AD-A065 480 UNG (MAN T) EL SEGUNDO CALIF F/G 9/2

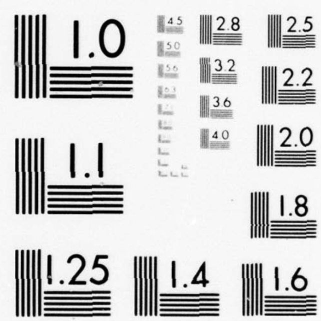
SPACE SHUTTLE DIGITAL COMPUTER SIMULATION BENCHMARK.(U)

DEC 78 M T UNG F04700-78-M-2539

UNCLASSIFIED AFFTC-TR-78-30 NL

1 OF 2

AD A065480



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

LEVEL II

(2)

AFFTC-TR-78-30

AD A065480

DDC FILE COPY

SPACE SHUTTLE DIGITAL COMPUTER SIMULATION BENCHMARK

Man T. Ung, PhD
428 W. Acacia
El Segundo CA 90245

December 1978

Final Report

This document has been approved for public release and resale; its distribution is unlimited.

Prepared for:
6510 TESTW/TEEST
Edwards AFB CA 93523

AIR FORCE FLIGHT TEST CENTER
EDWARDS AIR FORCE BASE, CALIFORNIA
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE

79 03 05 105

This report was submitted by Man T. Ung, PhD, 428 West Acacia, El Segundo, California 90245, under contract Number F04700-78-M-2539, Job Order Number 995030, with the Air Force Flight Test Center, Edwards AFB, California 93523. Mr Richard R. Hansen, Jr., was the AFFTC Project Engineer in charge of the procurement.

This report has been reviewed and cleared for open publication and/or public release by the AFFTC Office of Information in accordance with AFR 190-17 and DODD 5230.9. There is no objection to unlimited distribution of this report to the public at large, or by DDC to the National Technical Information Service where it may be released to foreign nationals.

Publication of this technical report does not constitute Air Force approval of its findings or conclusions. It is published only as a record of the technical effort and for the exchange and stimulation of ideas concerning aircraft flight simulation on digital computers.

Prepared by:

Man Ung

MAN T. UNG, PhD

This report has been reviewed and is approved for publication: December 1978

FOR THE COMMANDER

John L. Wesesky

JOHN L. WESESKY
Chief, Development Division

Alfred D. Phillips

ALFRED D. PHILLIPS
Technical Director
Test Engineering

When U.S. Government drawings, specifications, or other data are used for any purpose other than a definitely related government procurement operation, the government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded by implication or otherwise, as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

Do not return this copy; retain or destroy.

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: AFFTC TR-78-30	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): SPACE SHUTTLE DIGITAL COMPUTER SIMULATION BENCHMARK		5. TYPE OF REPORT & PERIOD COVERED: FINAL rept.
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): Man T. Ung, PhD		8. CONTRACT OR GRANT NUMBER(s): Contract F04700-78-M-2539 new
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Man T. Ung, PhD, 428 W. Acacia, El Segundo CA 90245		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 65807F
11. CONTROLLING OFFICE NAME AND ADDRESS		12. REPORT DATE: December 1978
		13. NUMBER OF PAGES: 164 (12) 172p.
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report): Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE: N/A

16. DISTRIBUTION STATEMENT (of this Report)

This document has been approved for public release and resale; its distribution is unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

N/A

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Aircraft Equations of Motion Simulation
Digital Flight Control System
Flight Control System Simulation
Space Shuttle Simulation
Simulator for Aircraft Flight Test and Development
Benchmark
Computer Simulation
Dynamic Check
Flight Simulation

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report was produced to serve as the benchmark to be used during the acceptance test for the Simulator for Flight Test and Development (SAFTD) at Edwards AFB. Afterwards, the document will be used to train new engineers to use the simulator. The writing is tutorial in nature and it stresses the orderly organization of a computer flight simulation. Model validation is explained in terms of static and dynamic checks. The report is not intended next page

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE 408 882 JOB

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

79 03 05 105

20. ABSTRACT (cont)

to arrive at any operational conclusion or new discoveries. This Space Shuttle Simulation, which is aimed at training and contractual acceptance of procured hardware, is different from the AFFTC Office of Advanced Manned Vehicles' Space Shuttle Simulation, which is used to support Shuttle development and flight testing.

PREFACE

The following document was intended to serve as a reference manual which will enable the reader to gain a quick understanding into the design and working arrangements of the Space Shuttle Entry and Approach Test simulation as used on the Simulator for Aircraft Flight Test and Development Simulation Benchmark. By reading through this report he is expected to know how to make use of the program and to find his way around in case any modification/addition is comtemplated. In brief, the main purpose for this report is to insure a smooth transferability from the originators to the users with a minimum period of learning and associated delay.

The program was written in Fortran for the SEL 32/55 digital computer with 48K 600 nsec memory equipped with floating-point hardware and tied to a functioning mock-up of an aircraft cockpit. Some part of this program was coded in Assembly language to improve the execution speed. Therefore the package cannot be transferred to another type of machine without some re-programming efforts.

Significant contributions to all phases of the simulation and report-writing were made by the following personnel:

Mr. Richard Hansen, Supervisor
Mr. Steve E. Louton, Engineer
Mr. Larry V. LeDuc, Engineer
Capt. Charles L. Bozeman, USAF
Lt. Douglas V. Palmer, USAF

TABLE OF CONTENTS

LIST OF TABLES AND FIGURES

1. INTRODUCTION

The Shuttle software package is a general-purpose real-time digital computer program that simulates all aspects of the flight: aerodynamics, flight control system connected to a cockpit for man-in-the-loop operation. The Space Shuttle mission profile is composed of many modes. They are very briefly: ignition & lift-off, pitch initialization after about 6 seconds, staging 122 seconds after lift-off (in the neighborhood of 235-nautical-mile altitude), external tank separation & disposal then orbit insertion. On the way back to earth the mission can be subdivided into: deorbit phase at 160-nautical-mile altitude, entry mode starting at 400,000ft, terminal phase starting at 50,000ft and finally landing and touchdown. The simulation contained in this report is devoted to the last two phases: Entry (beginning at 10^5 ft) and Terminal Area Energy Management (TAEM)

Handling quality constitutes the main objective of this study. It does not address the Shuttle's full mission nor does it simulate the navigation to pinpoint a landing site. Actual touchdown will not be simulated because the model does not include ground effects, the lowering of landing gear assembly and nose wheel steering. The Space Shuttle simulation described in this report is designed for training and contractual acceptance of procured simulator hardware. It is different from the Shuttle simulation used by the AFFTC Office of Advanced Manned Vehicles which is used for development and flight testing of the Shuttle.

2. MATHEMATICAL MODEL

By writing the aerodynamic equations, three explicit coordinate systems are used. They are H-frame (flight-path axes for describing orbital mechanics), body-axes and earth axes. The choice of various coordinate systems is made to enhance the relative computational accuracy of each segment of the model. For example, it makes no sense writing the vehicle rotational equations in any coordinate system except the ones with an origin residing within the vehicle itself such as body-axes, stability axes, or wind axes.

The transformation from body-axes to earth-axes is accomplished with a matrix composed of direction-cosine terms. These direction cosines are listed below but they are not explicitly evaluated in the program.

$$\ell_1 = \cos\theta \cos\psi$$

$$\ell_2 = \cos\theta \sin\psi \qquad (2\text{-}1)$$

$$\ell_3 = -\sin\theta$$

$$m_1 = -\cos\phi \sin\psi + \sin\phi \sin\theta \cos\psi$$

$$m_2 = \cos\phi \cos\psi + \sin\phi \sin\theta \cos\psi \qquad (2\text{-}2)$$

$$m_3 = \sin\phi \cos\theta$$

$$n_1 = \sin\phi \sin\psi + \cos\phi \sin\theta \cos\psi$$

$$n_2 = -\sin\phi \cos\psi + \cos\phi \sin\theta \sin\psi \qquad (2\text{-}3)$$

$$n_3 = \cos\theta \cos\phi$$

The angles θ, ϕ and ψ represent the pitch, roll and yaw angles. Together they are commonly referred to as the Euler angles.

For clarity, the model equations are arranged in groups , each group describing an aspect of the flight. Subscripts H, B, and E stand for the three coordinate systems mentioned earlier. Whatever symbol not explicitly named can be found in alphabetical order under Chapter 8 .

2.1 Aerodynamic equations: Acceleration terms in body-axis reference frame are the results of thrust vectors T_x, T_y, and T_z. The subscript "G" stands for landing gear and even though there is no engine, the thrust terms are included in the equations for completeness.

$$X_B = \frac{-\bar{q}S}{m}\left[C_{c_0} + \frac{C_{c_{(\delta e)_R}} C_{c_{(\delta e)_L}}}{2} + C_{c_{\delta BF}}(\delta BF) + C_{c_{\delta SB}}(\delta SB) + C_{c_G}(G)\right] + \frac{T_X}{m}$$

$$Y_B = \frac{\bar{q}S}{m}\{\ [C_{Y_{\delta r_{\delta SB}}}(\delta SB - \delta SB_{NOM}) + C_{Y_{\delta r}}]\ \delta r + C_{Y_{\delta_a}}(\delta a) + C_{Y_\beta}(\beta)\ \} + \frac{T_Y}{m} \qquad (2\text{-}4)$$

$$Z_B = -\frac{\bar{q}S}{m}[C_{N_0} + C_{N_{\delta e}}(\delta e) + C_{N_{\delta BF}}(\delta BF) + C_{N_{\delta SB}}(\delta SB - \delta SB_{NOM}) + C_{N_{\delta G}}(\delta G) + \Delta C_{N_{0(h)}}(C_{N_0})] + \frac{T_Z}{m}$$

Note that the subscript "N" refers to the normal force as opposed to subscript "n" which is associated with yawing-moment coefficients.

Resolved into the earth-axes reference frame, these acceleration terms become, after compensation for the earth's oblateness

$$X_E = X_B\ell_1 + Y_Bm_1 + Z_Bn_1 - 3J_2g_0\left(\frac{r_0}{r}\right)^4 \cos\bar{\mu}\sin\bar{\mu}$$

$$Y_E = X_B\ell_2 + Y_Bm_2 + Z_Bn_2 \qquad (2\text{-}5)$$

$$Z_E = X_B\ell_3 + Y_Bm_3 + Z_Bn_3$$

Where $\bar{\mu}$ and J_2 are the colatitude ($\frac{\pi}{2}$ - L) and the measure of earth oblateness respectively. The dynamic pressure $\bar{q}$ in the above equations and other related terms such as Mach number and the mass m are defined below.

$$\bar{q} = \frac{1}{2} \rho V^2$$

$$\text{Mach} = \frac{V}{a} \text{ ; } a = \text{sonic speed} \qquad (2\text{-}6)$$

$$m = \frac{W_0 + W_F}{g_0}$$

Here, W_0 and W_F stand for the vehicle empty weight and fuel weight respectively. Also the airspeed V, which is normally obtained by

$\sqrt{U_B^{\,2} + V_B^{\,2} + W_B^{\,2}}$ can simply be established by

$$V = \frac{U_B}{\cos\alpha \cdot \cos\beta} \qquad (2\text{-}7)$$

Latitude L and longitude λ are derived from the launching-point coordinate (L_0, λ_0) and the earth's rotation ω_e

$$L = L_0 + \int_0^t \frac{U_E}{r}\, dt$$

$$\lambda = \lambda_0 + \int_0^t \left[\frac{V_E}{r \cos L} - \omega_e \right] dt \qquad (2\text{-}8)$$

2.2 Orbital & Suborbital Equations: First consider the orbital phase. Let ψ_H be the angle between the direction of travel and true north and r the vehicle altitude measured from the earth center. Let (x_H, y_H) be the local plane perpendicular to the line of r (or z_H) such that x_H is pointing in the general direction of travel and y_H perpendicular to (x_H, z_H) to form a right-handed triad (see ref. 9.1.3) Then

$$rU_H = (rU_H)_0 + \int_0^t rX_H dt \qquad (2\text{-}9)$$

$$\psi_H = \psi_{H_0} + \int_0^t \left(\frac{+V_E}{r} \tan L + \frac{Y_H}{U_H} \right) dt$$

$$W_H = W_{H_O} + \int_0^t \left[g + Z_H \right] dt$$

$$\delta R = \delta R_0 + \int_0^t (-W_H) dt$$

$$U_H = \frac{(rU_H)}{r}$$

Where

$$r = r_0 + \delta R \quad (\delta R \text{ measured from sea level})$$

$$X_H = X_E \cos \psi_H + Y_E \sin \psi_H$$

$$Y_H = -X_E \sin \psi_H + Y_E \cos \psi_H \qquad (2-10)$$

$$Z_H = Z_E$$

And the velocity and the gravity terms are obtained as follows

$$U_E = U_H \cos \psi_H$$

$$V_E = U_H \sin \psi_H \qquad (2-11)$$

$$W_E = W_H$$

$$g = g_0 \left(\frac{r_0}{r}\right)^2 - \left[\frac{3}{2} J_2 g_0 \left(\frac{r_0}{r}\right)^4 (3 \cos^2 \bar{\mu} - 1)\right] - \frac{(U_H)^2}{r}$$

Next, consider the suborbital phase

$$U_B = U_{EA} \ell_1 + V_{EA} \ell_2 + W_E \ell_3$$

$$V_B = U_{EA} m_1 + V_{EA} m_2 + W_E m_3 \qquad (2-12)$$

$$W_B = U_{EA} n_1 + V_{EA} n_2 + W_E n_3$$

where the effective airspeeds U_{EA} and V_{EA} are defined in terms of the north (W_u) and east (W_v) component of the wind shears. No updraft/downdraft are included in this model. Values of W_u and W_v are stored at 5,000 foot increments, from 0 to 100,000 feet altitude where this simulation starts.

$$U_{EA} = U_E - W_u$$
$$V_{EA} = V_E - (r \cos L)\omega_e - W_v \qquad (2\text{-}13)$$

2.3 Rotational Equations: The moments acting on the vehicle are estimated about the body axes as shown below. Let ℓ_r be the reference length of the vehicle. Let J_R, J_P and J_Y be the dimensionless numbers of roll, pitch and yaw jet firings and L_R and L_Y be the rolling moments **due to one roll and one yaw reaction jet firing, respectively. Then**

$$L_B = \frac{\bar{q}Sb^2}{2V}\left[C_{\ell_p}(P_B) + C_{\ell_R}(R_B)\right] + \bar{q}Sb\left\{\left[C_{\ell_\beta} + C_{\ell_{\beta_{\delta e}}}(\delta e) + \left(C_{\ell_{\beta_{\delta SB_1}}} + C_{\ell_{\beta_{\delta SB_2}}}\right)(\delta SB - 25°) + C_{\ell_{\beta_{\delta SB_3}}}(\delta SB - 60°) + C_{\ell_{\beta_{\delta G}}}(\delta G)\right](\beta) + C_{\ell_{\delta a}}(\delta a) + C_{\ell_{\delta r}}(\delta r)\right\}$$
$$+\ m\cdot\ell_r\left[(z_{NOM} - \bar{z})\cdot Y_B - \bar{y}\cdot Z_B\right] + (L_R)\,J_R \qquad (2\text{-}14)$$

$$M_B = \frac{\bar{q}S\bar{c}^2}{2V}(C_{m_C})(C_L) + \bar{q}S\bar{c}\left[C_{m_0} + \frac{C_{m(\delta e)RIGHT} + C_{m(\delta e)LEFT}}{2} + C_{m_{\delta SB}}(\delta SB - \delta SB_{NOM}) + C_{m_{\delta BF}}(\delta BF) + C_{m_{\delta G}}(\delta G) + \Delta C_{m_e}\right] + m\cdot\ell_r\left[-(z_{NOM} - \bar{z})\cdot X_B + (x_{NOM} - \bar{x})\cdot Z_B\right] + (L_P)\,J_P \qquad (2\text{-}15)$$

$$N_B = \frac{\bar{q}Sb^2}{2V} [C_{n_p} (P_B) + C_{n_R} (R_B)] + \bar{q}Sb\{ [C_{n_{\beta_{\delta e}}} (\delta e)$$

$$+ (C_{n_{\beta_{\delta SB_1}}} + C_{n_{\beta_{\delta SB_2}}}) (\delta SB - 25°) + C_{n_{\beta_{\delta SB_3}}} (\delta SB - 60°)$$

$$+ C_{n_{\beta_{\delta G}}} (\delta G) + C_{n_\beta}](\beta) + C_{n_{\delta a}} (\delta a) + [C_{n_{\delta r}} \tag{2-16}$$

$$+ C_{n_{\delta r_\beta}} (|\beta|)] (\delta r)\} + (\bar{y}C_B - \bar{x}Y_B) (m) (\ell_r) + (L_Y) J_Y$$

Appropriate reaction-jet terms can be added to the M_B and N_B equations if later vehicles are so equipped. The mean aerodynamic chord length is represented by $\bar{c}$ in the above equation. The symbol $\bar{x}$ represents the x-axis displacement of center-of-gravity from an arbitrarily picked reference point. Similar definitions apply to $\bar{y}$ and $\bar{z}$.

$$\bar{x} = (cg_{X_0} - cg_{X_{NOM}}) + \left(\frac{W_F}{W_{F_{max}}}\right) \Delta\bar{x}_F + \frac{\sin \theta_F}{|\sin \theta_F|} (\Delta\bar{x}_{\theta_F}) \sqrt{\sin |\theta_F|}$$

$$\bar{y} = (cg_{Y_0} - cg_{Y_{NOM}}) \tag{2-17}$$

$$\bar{z} = (cg_{Z_0} - cg_{Z_{NOM}}) + \left(\frac{W_F}{W_{F_{max}}}\right) \bar{z}_F$$

Here, θ_F **the angle made by the surface of the fuel with respect to the $\bar{X}_B$ axis**

$$\theta_F = 90° + \tan^{-1} \frac{Z_B}{X_B} \tag{2-18}$$

Using the moments L_B, M_B, and N_B, we can obtain the vehicle angular accelerations about its own axes - - $\dot{P}_B$, $\dot{Q}_B$ and $\dot{R}_B$

$$\dot{P}_B = \frac{1}{I_{XX}} [L_B - I_{XY}(P_B R_B - \dot{Q}_B) + I_{XZ}(P_B Q_B + \dot{R}_B)$$

$$+ I_{YZ}({Q_B}^2 - {R_B}^2) + Q_B R_B (I_{YY} - I_{ZZ})]$$

$$\dot{Q}_B = \frac{1}{I_{YY}} [M_B + I_{XY}(Q_B R_B + \dot{P}_B) - I_{XZ}({P_B}^2 - {R_B}^2)$$

(2 -19)

$$- I_{YZ}(P_B Q_B - \dot{R}_B) + P_B R_B (I_{ZZ} - I_{XX})]$$

$$\dot{R}_B = \frac{1}{I_{ZZ}} [N_B - I_{XY}({Q_B}^2 - {P_B}^2) - I_{XZ}(Q_B R_B - \dot{P}_B)$$

$$+ I_{YZ}(P_B R_B + \dot{Q}_B) + P_B Q_B (I_{XX} - I_{YY})]$$

where the inertia terms are defined as follows

$$I_X = I_{XX} = I_{XX_O} = \left(\frac{W_F}{W_{F_{max}}}\right) I_{XX_F}$$

$$I_Y = I_{YY} = I_{YY_O} + \left(\frac{W_F}{W_{F_{max}}}\right) I_{YY_F}$$

(2 -20)

$$I_Z = I_{ZZ} = I_{ZZ_O} + \left(\frac{W_F}{W_{F_{max}}}\right) I_{ZZ_F}$$

$$I_{XZ} = I_{XZ_O} + \left(\frac{W_F}{W_{F_{max}}}\right) I_{XZ_F}$$

$$I_{XY} = I_{XZ_O} + \left(\frac{W_F}{W_{F_{max}}}\right) I_{XY_F}$$

$$I_{YZ} = I_{YZ_O} + \left(\frac{W_F}{W_{F_{max}}}\right) I_{YZ_F}$$

Fuel consumption and the weight of fuel are described by

$$\dot{W}_F = -(F_T)T \quad ; \quad T = \text{Thrust}$$

$$W_F = W_{F0} + \int_0^t \dot{W}_F dt \qquad (2\text{-}21)$$

By integrating the angular accelerations we acquire the angular velocities P_B, Q_B and R_B (also known as the rolling rate, pitching rate and yawing rate).

$$P_B = P_{B_0} + \int_0^t \dot{P}_B dt$$

$$Q_B = Q_{B_0} + \int_0^t \dot{Q}_B dt \qquad (2\text{-}22)$$

$$R_B = R_{B_0} + \int_0^t \dot{R}_B dt$$

2.4 Other Pertinent Equations: Euler angles namely pitch, yaw and roll, are calculated below. Notice that this formulation will not tolerate a pitch angle θ of $\pm 90°$.

$$\theta = \theta_0 + \int_0^t (Q_B \cos\phi - R_B \sin\phi) dt$$

$$\phi = \phi_0 + \int_0^t [P_B + \tan\theta(R_B \cos\phi + Q_B \sin\phi)]\, dt \qquad (2\text{-}23)$$

$$\psi = \psi_0 + \int_0^t \left(\frac{R_B \cos\phi + Q_B \sin\phi}{\cos\theta}\right) dt$$

Stability-axis angles, namely angle of attack α and angle of sideslip β can be computed as

$$\alpha = \tan^{-1}\left(\frac{W_B}{U_B}\right)$$

$$\beta = \tan^{-1}\left(\frac{V_B \cos\alpha}{U_B}\right) \qquad (2\text{-}24)$$

Flight path angle

$$\gamma = \theta - \alpha \cos\phi - \beta \sin\phi \qquad (2\text{-}25)$$

Let ψ_R be the given angle between true north and the R_D vector; then down range and cross range can be computed as:

$$R_D = r_0 \left[(L - L_r)\cos\psi_R + (\lambda - \lambda_0)\cos L \sin\psi_R\right]$$

$$R_X = r_0 \left[(L - L_r)\sin\psi_R + (\lambda - \lambda_0)\cos L \cos\psi_R\right] \qquad (2\text{-}26)$$

Equivalent airspeed KEAS and calibrated airspeed KCAS

$$\text{KEAS} = \left(V\sqrt{\frac{\rho}{\rho_0}}\right)\frac{3600}{6080} \quad \text{nautical miles}$$

$$\text{KCAS} = \left(\frac{\nu_1}{\nu_2}\right) \cdot (\text{KEAS}) \qquad (2\text{-}27)$$

The quantities ν_1 and ν_2 are defined by the following empirical formulae:

$$\nu_1 = f(x) \quad ; \quad x = \text{Mach number} \qquad (2\text{-}28)$$

$$\nu_2 = f(x) \quad ; \quad x = (.001511)\,(\text{KCAS})$$

Where f(x) assumes the following form:

$$f(x) = \begin{cases} \sqrt{1 + \frac{x^2}{4} + \frac{x^4}{40} + \frac{x^6}{1600}} & \text{for } x < 1.0 \\ \sqrt{1.839 - \frac{.772}{x^2} + \frac{.164}{x^4} + \frac{.035}{x^6}} & \text{for } x \geq 1.0 \end{cases} \tag{2-29}$$

Geometric altitude h and geopotential altitude:

$$h = \delta r + (\sin^2 L)\,\delta h$$

$$h_p = \frac{(r_0)(h)}{r_0 + h} \tag{2-30}$$

Rate of climb:

$$\frac{d}{dt}(h) = -W_H = -W_E \tag{2-31}$$

Air density can be approximated as a function of altitude, based upon standard atmospheric data:

$$\rho = \rho_0 E^{-\zeta h_p} \tag{2-32}$$

3. MAIN PROGRAM

No efforts are spared to organize the main program in a self-explanatory manner. Parameters and data are grouped together according to their roles and the omnipresence of the comment cards should guide the reader through the program. It is suggested that subsequent users do not deviate from the existing logical organization when making modifications/additions, so not to defeat the purposes. In order to simplify the loading process, all parameters and data (including aerodynamic coefficients) are stored in the main program , thus avoiding the use of COMMON blocks.

As indicated by figure 3 -1, the main program is composed of three parts: the initialization, the Hi-speed loop (starting from statement 47), and the EASE subroutine designed to display and modify any parameter of the main program. EASE was written to allow interchange of parameters without the use of COMMON blocks.

3.1 Initialization: Most data parameters and initial conditions are on punched cards to allow maximal flexibility on flight conditions and vehicle configurations. All aerodynamic derivatives are stored in the form of 16-bit normalized fixed point fractions. The detailed discussion of their handling will be covered under the heading of "Function Generation". During the initialization phase the program will ask the user if he is through with changing parameters before entering into the next phase, by typing the question "ARE FURTHER CHANGES DESIRED?" A "YES" answer will automatically call in the EASE program. A "NO" answer will set in motion the high-speed loop.

Not all parameters can be independently initialized. For instance, V_E is called an "intermediate" variable because it is a function of ψ_H and U_H. Thus the initial value of V_E is determined once the initial conditions of ψ_H and U_H are specified, and vice versa.

3.2 Hi-Speed Loop: This loop is actuated by an interrupt which occurs at synchronous, regular intervals Δt whose duration was set a-priori. During the time Δt, the computer must finish solving one frame of the dynamic equations before the occurrence

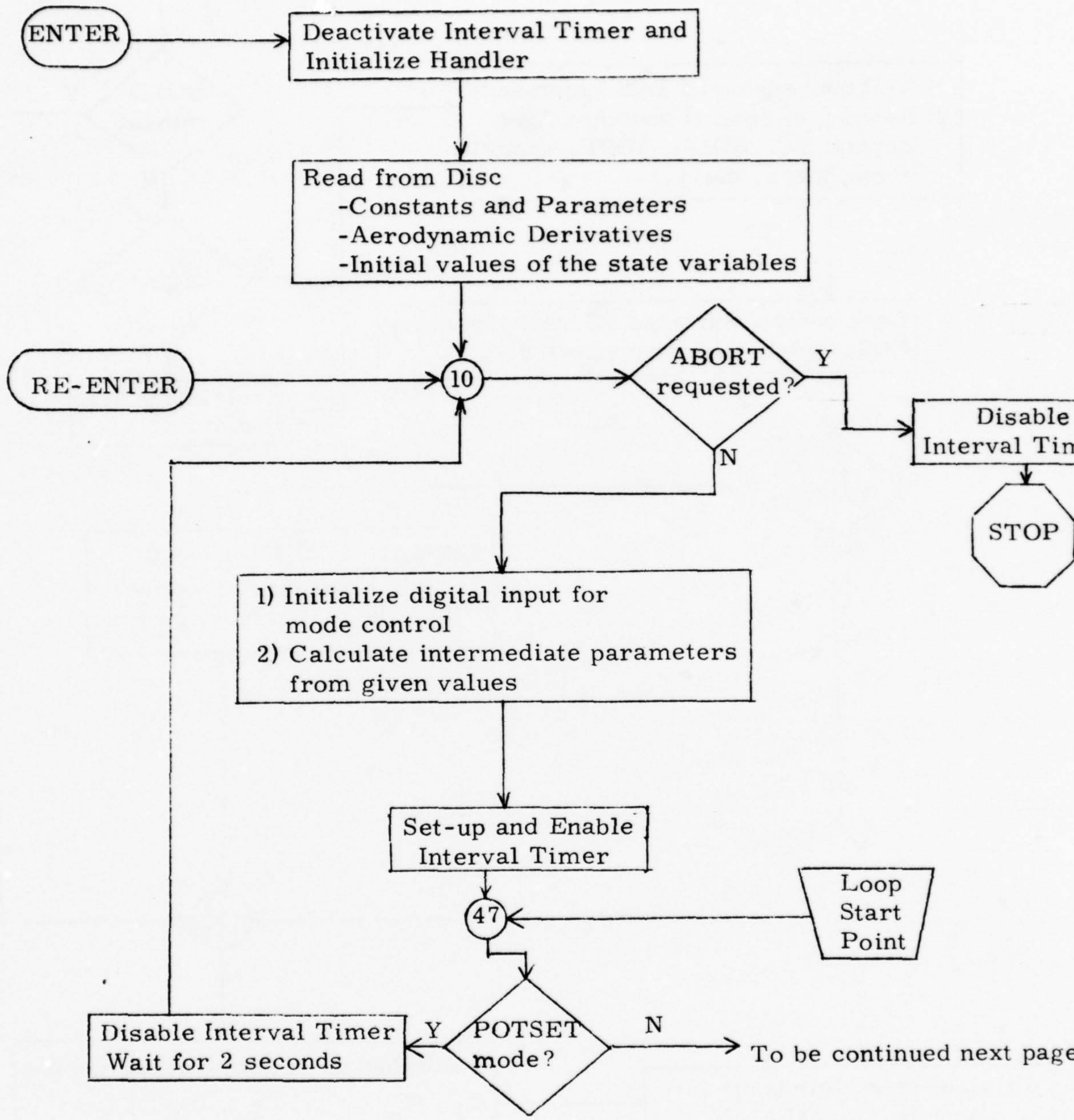
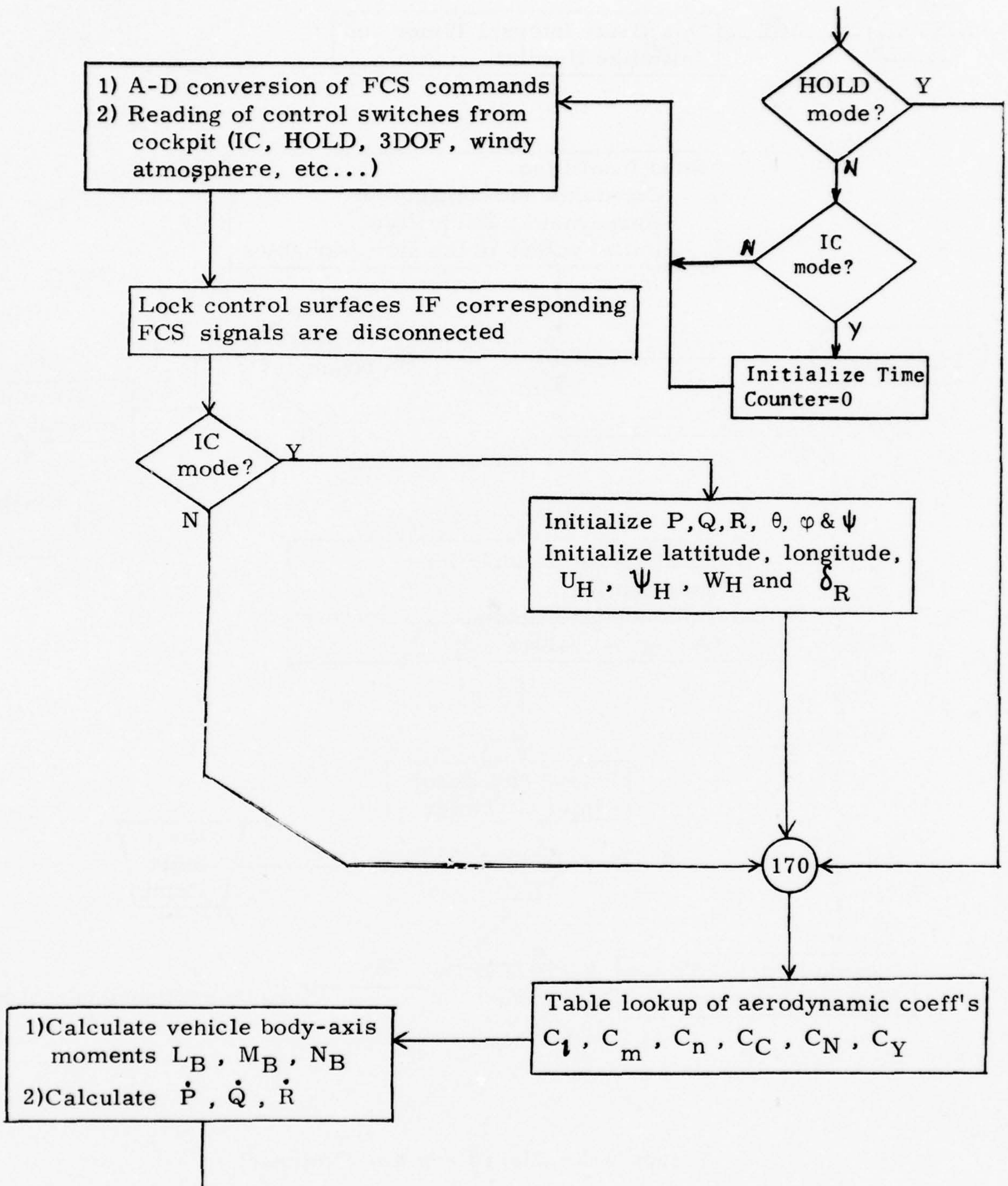


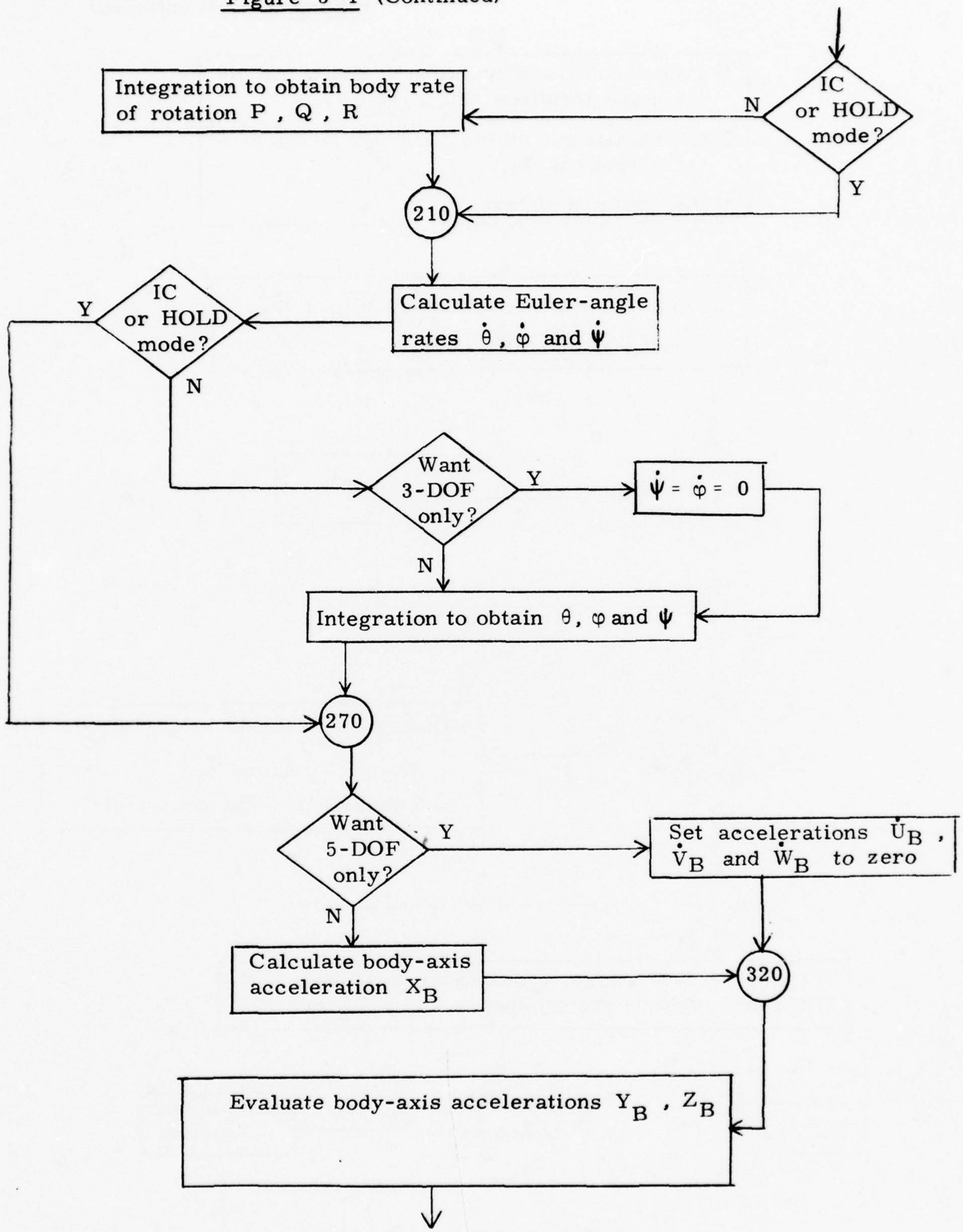
Figure 3-1 - Main Program Flowchart

Figure 3-1 (Continued)



To Be Continued
On Next Page

Figure 3-1 (Continued)



To Be Continued On Next Page

Figure 3-1 (Continued)

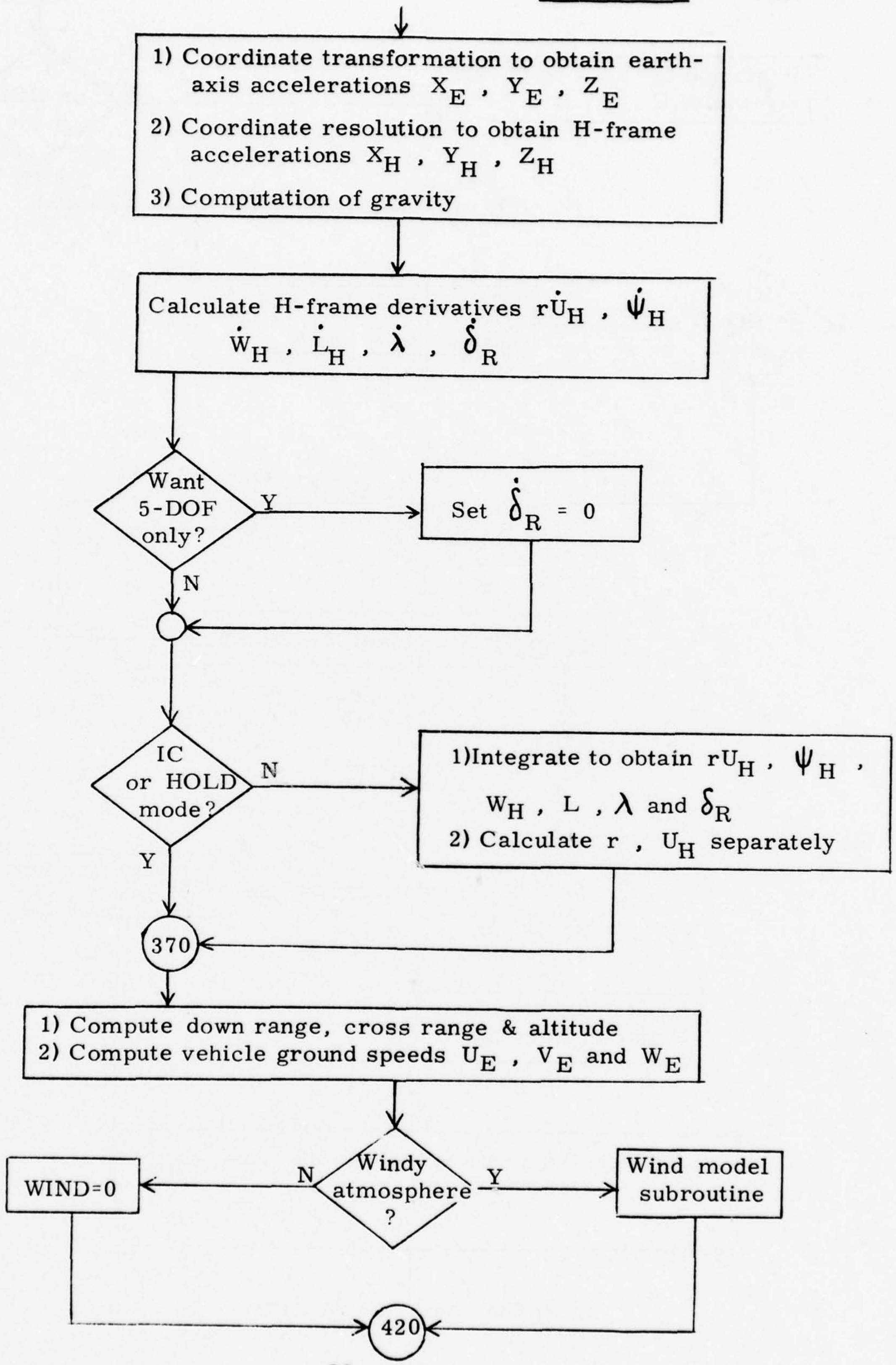
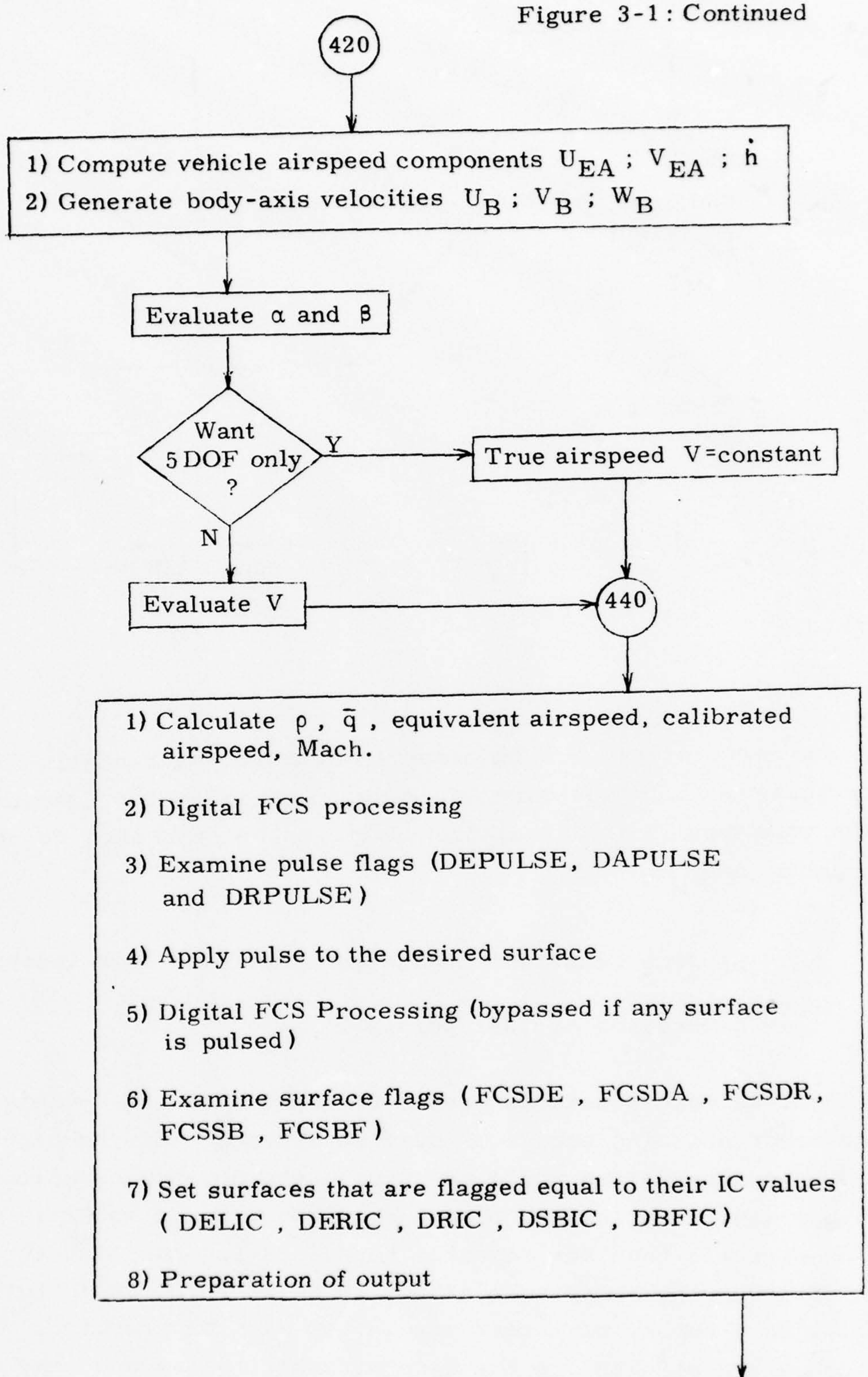
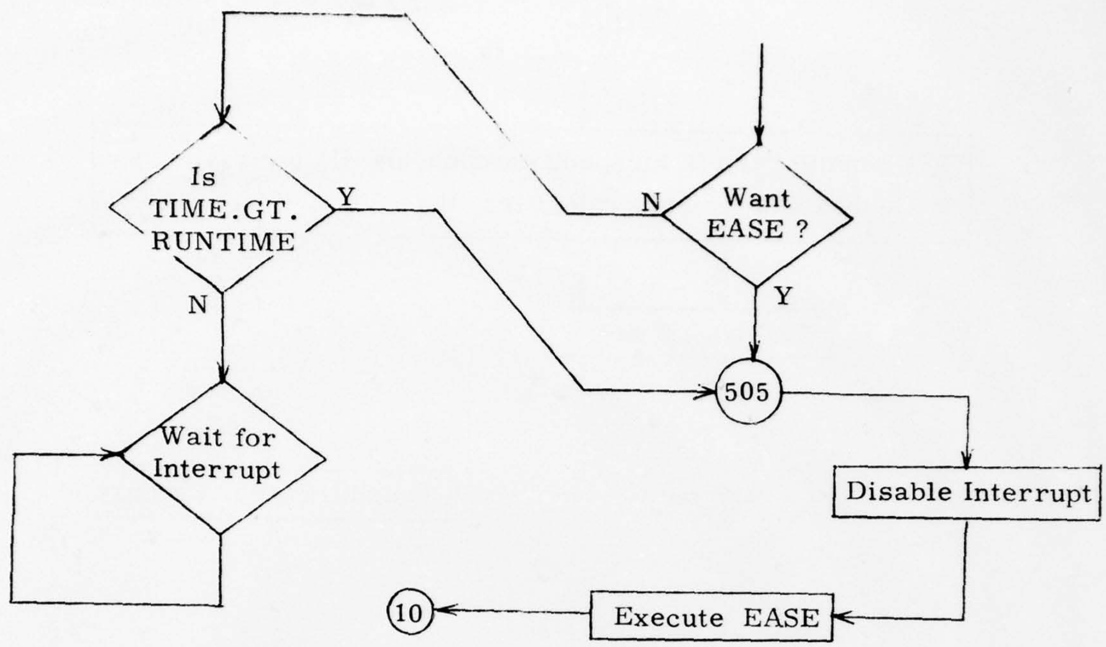


Figure 3-1: Continued



To be continued on next page

Figure 3-1: Continued



of the next interrupt. In order to achieve the real-time (or time-critical) goal, great care was made to minimize the execution time. Even so, a number of calculations have to be delegated to the analog computer such as

- Sin/cos functions of angles θ, ϕ, ψ, α for instrumentation
- generation of test pulses

In fact, if more equations are to be added into the dynamic model, some other existing equations must be deleted from the high-speed loop. An alternative would be solving the deleted equations on another digital or analog processor. The dynamic model is given in Chapter II, thus not repeated here. During run time, the operator can intervene by throwing a number of switches which in turn activiate a number of interrupts in the digital computer. The switches are situated in the general cockpit area and they consist of

- IC Switch: sets the program back to Initial Condition mode. All information concerning the preceding run is lost.

- HOLD Switch: also stops execution but all state variable values are frozen at their respective values reached prior to activation of the switch. This mode allows the user to investigate the status of his program in post-mortem fashion.

- 3 DOF Switch: sets the yaw rate $\dot{\psi}$ and the roll rate $\dot{\theta}$ to zero, the vehicle can then be studied entirely in the pitch plane, having three degrees-of-freedom: $\vec{x}$, $\vec{z}$, and θ. This mode is useful in planning and performance analysis such as plotting the gliding envelope. In this case, we are interested only in getting from one point to another point.

- 5 DOF Switch: imposes a nearly constant altitude, constant total-velocity flight regime . This mode offers a convenient way to study vehicle handling quality and maneuverability. To different people, 5-DOF simulation means slightly different things. Thus we derive the 5-DOF situation used in this simulation for reference purposes.

Consider the acceleration equations in body-axes, excluding the thrust and aerodynamic force on the surfaces.

$$\begin{bmatrix} X_B \\ Y_B \\ Z_B \end{bmatrix} = \begin{bmatrix} \dot{U}_B \\ \dot{V}_B \\ \dot{W}_B \end{bmatrix} + \begin{bmatrix} 0 & W_B & -V_B \\ -W_B & 0 & U_B \\ V_B & -U_B & 0 \end{bmatrix} \begin{bmatrix} P_B \\ Q_B \\ R_B \end{bmatrix}$$

$$
+ \underbrace{\begin{bmatrix} c\psi.c\theta & s\psi.c\theta & -s\theta \\ (c\psi.s\theta.s\phi - s\psi.c\phi) & (s\psi.s\theta.s\phi + c\psi.c\phi) & c\theta.s\phi \\ (c\psi.s\theta.c\phi + s\psi.s\phi) & (s\psi.s\theta.c\phi - c\psi.s\phi) & c\theta.c\phi \end{bmatrix}}_{\text{Inertial-axis} \rightarrow \text{body-axis transformation}} \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} \qquad (3\text{-}1)
$$

Where $c\psi$ and $s\psi$ are the abbreviations for cos ψ and sin ψ respectively. For 5DOF, assume that $\dot{U}_B = \dot{V}_B = \dot{W}_B = 0$ and solve for the acceleration component X_B

$$
X_B = - \frac{Z_B W_B + Y_B V_B}{U_B} + g \left[\sin\theta - \frac{\cos\theta (V_B \sin\phi - W_B \cos\phi)}{U_B} \right] \qquad (3\text{-}2)
$$

Equation (3 -2) is used in lieu of X_B described in equations (2 -4). The only other changes in the 5-DOF case are

$$
\frac{d(\Delta r)}{dt} = 0 \qquad (3\text{-}3)
$$

and $V = V_{IC}$, a constant

The closed-loop nature of the hi-speed loop is illustrated by the block diagram of figure 3 -2. This block diagram depicts the main stream of the calculations and how different groups of calculations are related to each other functionally.

Figure 3-2: Block Diagram

Function Generation

Thrust, reaction jets

control surface deflections

Flight Control Systems

H - Axis Accelerations

$$X_H = X_E \cos\psi_H + Y_E \sin\psi_H$$

$$Y_H = -X_E \sin\psi_H + Y_E \cos\psi_H$$

$$Z_H = Z_E$$

Euler-Axis Accelerations

X_B: $$X_E = X_B \ell_1 + Y_B m_1 + Z_B n_1 - 3J_2 g_o \left(\frac{r_o}{r}\right)^4 \cos\bar{\mu} \sin\bar{\mu}$$

Y_B: $$Y_E = X_B \ell_2 + Y_B m_2 + Z_B n_2$$

Z_B: $$Z_E = X_B \ell_3 + Y_B m_3 + Z_B n_3$$

Winds and Rotating Earth

$$U_{EA} = U_E - W_U$$

$$V_{EA} = V_E - W_V - (r\cos L)\omega_e$$

Directional Cosines

$$\ell_1 = \cos\theta \cos\psi$$

$$\ell_2 = \cos\theta \sin\psi$$

$$\ell_3 = -\sin\theta$$

$$m_1 = -\cos\phi \sin\psi + \sin\phi \sin\theta \cos\psi$$

$$m_2 = \cos\phi \cos\psi + \sin\phi \sin\theta \sin\psi$$

$$m_3 = \sin\phi \cos\theta$$

$$n_1 = \sin\phi \sin\psi + \cos\phi \sin\theta \cos\psi$$

$$n_2 = -\sin\phi \cos\psi + \cos\phi \sin\theta \sin\psi$$

$$n_3 = \cos\theta \cos\phi$$

Body Axis Rotational Accelerations

L_B: $$\dot{P}_B = \frac{1}{I_X}\left[L_B - I_{XY}(P_B R_B - \dot{Q}_B) + I_{XZ}(P_B Q_B + \dot{R}_B) + (Q_B R_B)(I_Y - I_Z)\right]$$

M_B: $$\dot{Q}_B = \frac{1}{I_Y}\left[M_B + I_{XY}(Q_B R_B + \dot{P}_B) - I_{XZ}(P_B^2 - R_B^2) + (P_B R_B)(I_Z - I_X)\right]$$

N_B: $$\dot{R}_B = \frac{1}{I_Z}\left[N_B + I_{XY}(Q_B^2 - P_B^2) - I_{XZ}(Q_B R_B - \dot{P}_B) + (P_B Q_B)(I_X - I_Y)\right]$$

P_B

Q_B

R_B

am

Latitude, Longitude, Altitude

$$L = L_0 + \int_0^t \frac{V_E}{r}\,dt$$

$$\lambda = \lambda_0 + \int_0^t \left(\frac{V_E}{r \cos L} - \omega_e\right)dt$$

$$\delta r = \delta r_0 + \int_0^t (-W_H)\,dt$$

$$h = \delta r + (\sin^2 L)\,\delta h \;;\; r = r_0 + \delta r$$

Gravity

$$g = g_0\left(\frac{r_0}{r}\right)^2 - \frac{(U_H)^2}{r}$$

$$-\left[\frac{3}{2} J_2 g_0 \left(\frac{r_0}{r}\right)^4 (3\cos^2\bar{\mu} - 1)\right]$$

H - Axis Velocities

$$rU_H = \left(rU_{H_0}\right) + \int_0^t rX_H\,dt$$

$$\psi_H = \psi_{H_0} + \int_0^t \left[\frac{V_E}{r}\tan L + \frac{Y_H}{U_H}\right] dt$$

$$W_H = W_{H_0} + \int_0^t \left[g + Z_H\right] dt$$

$$U_H = \frac{\left(rU_H\right)}{r}$$

tions

$\sin\psi_H$

$\cos\psi_H$

Earth-Axis Velocities

$$U_E = U_H \cos \psi_H$$

$$V_E = U_H \sin \psi_H$$

$$W_E = W_H = \text{-HDOT}$$

otating Earth

$_E - W_U$

$_E - W_V$

$r\cos L)\,\omega_e$

Airspeed, Angles of attack & sideslip

$$\alpha = \tan^{-1} \frac{W_B}{U_B}$$

$$\beta = \tan^{-1} \frac{V_B \cos \alpha}{U_B}$$

$$V = \sqrt{U_B^{\,2} + V_B^{\,2} + W_B^{\,2}}$$

Body-Axis Velocities

$$U_B = U_{EA}\ell_1 + V_{EA}\ell_2 + W_E\ell_3$$

$$V_B = U_{EA}m_1 + V_{EA}m_2 + W_E m_3$$

$$W_B = U_{EA}n_1 + V_{EA}n_2 + W_E n_3$$

$$\rho = \rho_0 \varepsilon^{-\xi h_\rho}$$

$$\bar{q} = \frac{1}{2}\rho V^2$$

$$\text{Mach} = V / a$$

Euler Angles

$$\theta = \theta_0 + \int_0^t (Q_B \cos\phi - R_B \sin\phi)\,dt$$

$$\phi = \phi_0 + \int_0^t \left[P_B + \tan\theta\,(R_B \cos\phi + Q_B \sin\phi)\right] dt$$

$$\psi = \psi_0 + \int_0^t \frac{(R_B \cos\phi + Q_B \sin\phi)}{\cos\theta}\,dt$$

Body Axis Rotation Rates

$$P_B = P_{B_0} + \int_0^t \dot{P}_B\,dt$$

$$Q_B = Q_{B_0} + \int_0^t \dot{Q}_B\,dt$$

$$R_B = R_{B_0} + \int_0^t \dot{R}_B\,dt$$

$I_Z)]$

$I_X)]$

$I_Y)]$

2

2

4. FLIGHT CONTROL SYSTEM (FCS)

4.1 Description: In this simulation we investigate the Orbiter handling qualities under manual control of a pilot. There is no provision for an autopilot in this program. From the time the Orbiter enters the atmosphere at a nominal altitude of 400,000ft to touchdown it is not power-assisted except for the Reaction-jet Control System (RCS). Thus the importance of the FCS can't be too much exagerated.

The Orbiter is equipped with an all-digital fly-by-wire FCS which insures aerodynamic stability with the help of five on-board computers (four redundant primary and one secondary). The computers, in turn, feed signals to the actuators of the control surfaces . A picture of the control surfaces can be found on page 22 of reference 9.2.2.

a/ Elevons (Elevator-aileron combination): They are full-span in construction, used for affecting both pitch and roll. This program does not exercise the inboard and outboard elevons separately.

b/ Rudder / Speedbrakes: The conventional rudder is made up of two symmetrical halves which, when deployed, constitute the speed brakes.

c/ Body flap is found on the very end of the fuselage below the main engines and serves mainly as augmentation to longitudinal trim.

Whenever the control surfaces prove to be ineffective (high altitude, low dynamic pressures) pitch and roll reaction-jets are fired to guaranty the orbiter's aerodynamic stability. The yaw reaction jets are used more often, including the cases just mentioned above.

4.2 FCS Block Diagrams: The pitch, roll and yaw modes of the FCS are depicted by the block diagrams on figures 4-1 , 4-2 and 4-3 in that order. Let us discuss one of the modes in detail, say the pitch mode of figure 4-1 . A command from the RHC (Rotational Hand Command) is fed through the deadband and pitch shaping networks then compared with the current Orbiter performance (a function of the pitch rate Q) to generate a pitch rate error signal called DPJ. Because of the coupling effects between the axes, the nose tends to dip during a turn. To compensate for this, we want an additional negative elevon deflection (trailing edge up , by convention) and the term $R^*\tan\varphi$ does exactly this by contributing a negative influence to the elevator command $\delta_{e_{CMD}}$.

When the speedbrakes are deployed, they force the nose upward by contributing a moment around the Orbiter's center of gravity (CG). Note how the speedbrake increment (DSBPC) signal is used to negate the unwanted pitch-up by a direct reduction in the pitch-position trim.

The FSC roll channel is represented on figure 4-2. The switches EARLY and LATE correspond to MACH >1.5 and MACH ≤ 1.5 respectively. Notice the roll-rate command main path. Starting with the roll stick position, the signal goes through deadband, shaping and first-order lag circuits before being added with the integral of the roll-rate trim term at SUM21. From there, the control signal is sent through SUM 24 where the roll-rate feedback and other coupling terms are subtracted. All the feedback signals taken together are called PSTAB (roll-rate stability). SUM24 is directed to a switch whose poles are marked EARLY and LATE. The EARLY side of the switch allows only reaction jets to control the roll channel because, at that time, the Orbiter is still at high altitude. Only during the LATE stage can the control surfaces perform effectively.

The FCS yaw channel is illustrated on figure 4-3. The rudder pedal generates a signal which is fed through the dead-band and shaping networks. Then it is summed with the integrated yaw-trim signal. Before reaching the actuators proper, this signal is further modified by Mach number and dynamic pressure $\bar{q}$. An arrangement similar to the roll-axis channel is implemented here to allow the yaw jets UZCMD to take over at high altitude (EARLY). Only at low altitude (LATE) can the rudder work the way it works on an airplane.

Figure 4-4 describes how the Orbiter's control surfaces are exercised by different actuators.

4.3 FCS Computer Implementation: In the block diagrams the transfer functions and filters are expressed in the Laplace s-domain. We could have used the s-plane if an analog computer were used. Since the FCS program is done digitally it is advantageous to map the transfer functions from the s-plane to another complex plane, the z-plane. In simple terms, z can be defined as $z=e^{sT}$ where T is the sampling period. A comprehensive treatment of infinite-impulse-response digital filters can be found in reference 9.1.6 , chapter 4.

It is a known fact that a stable analog filter might or might not map into a stable digital filter if we just go about replacing the differentials by finite

FIGURE 4–1: FCS PITCH CHANNEL

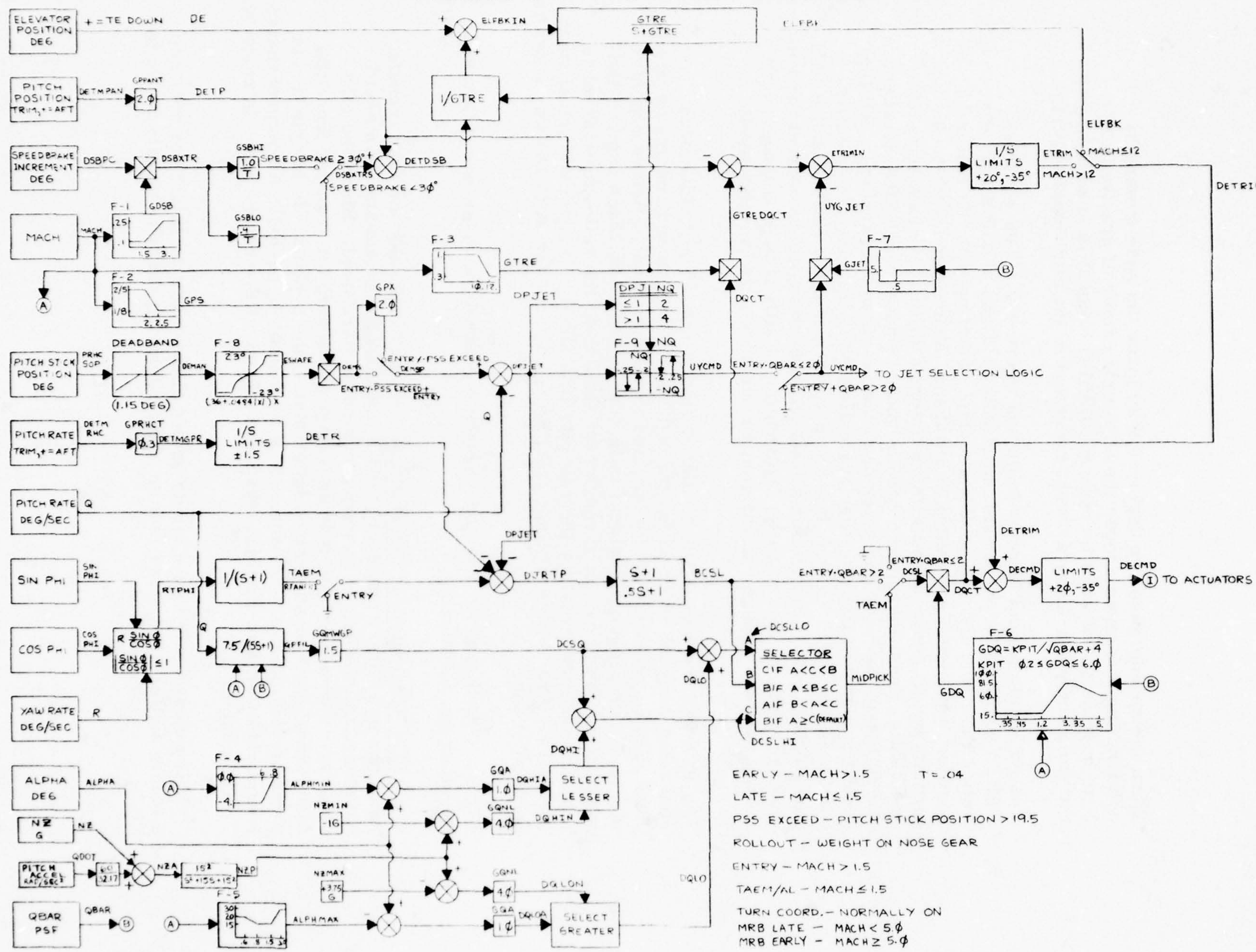
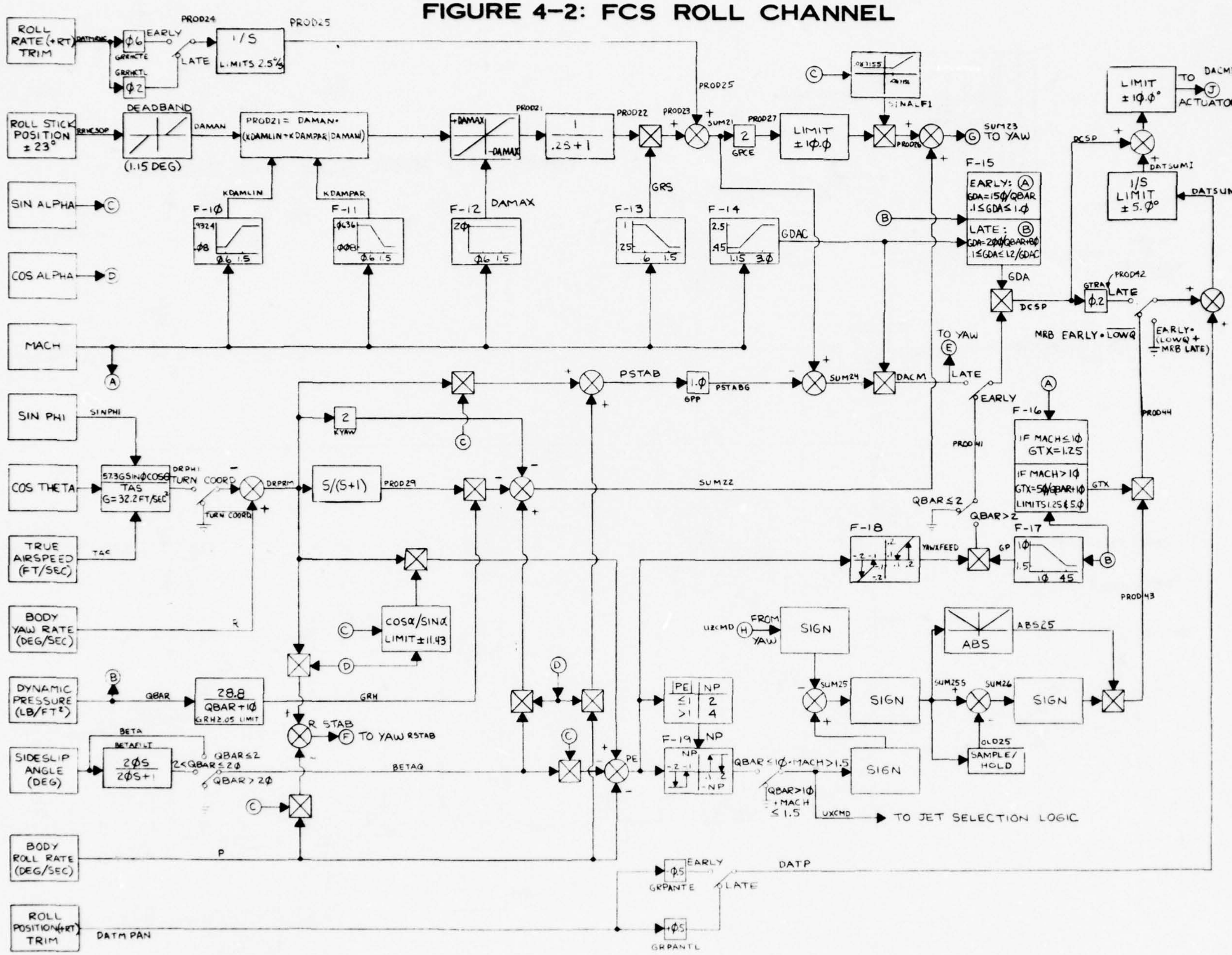


FIGURE 4-2: FCS ROLL CHANNEL



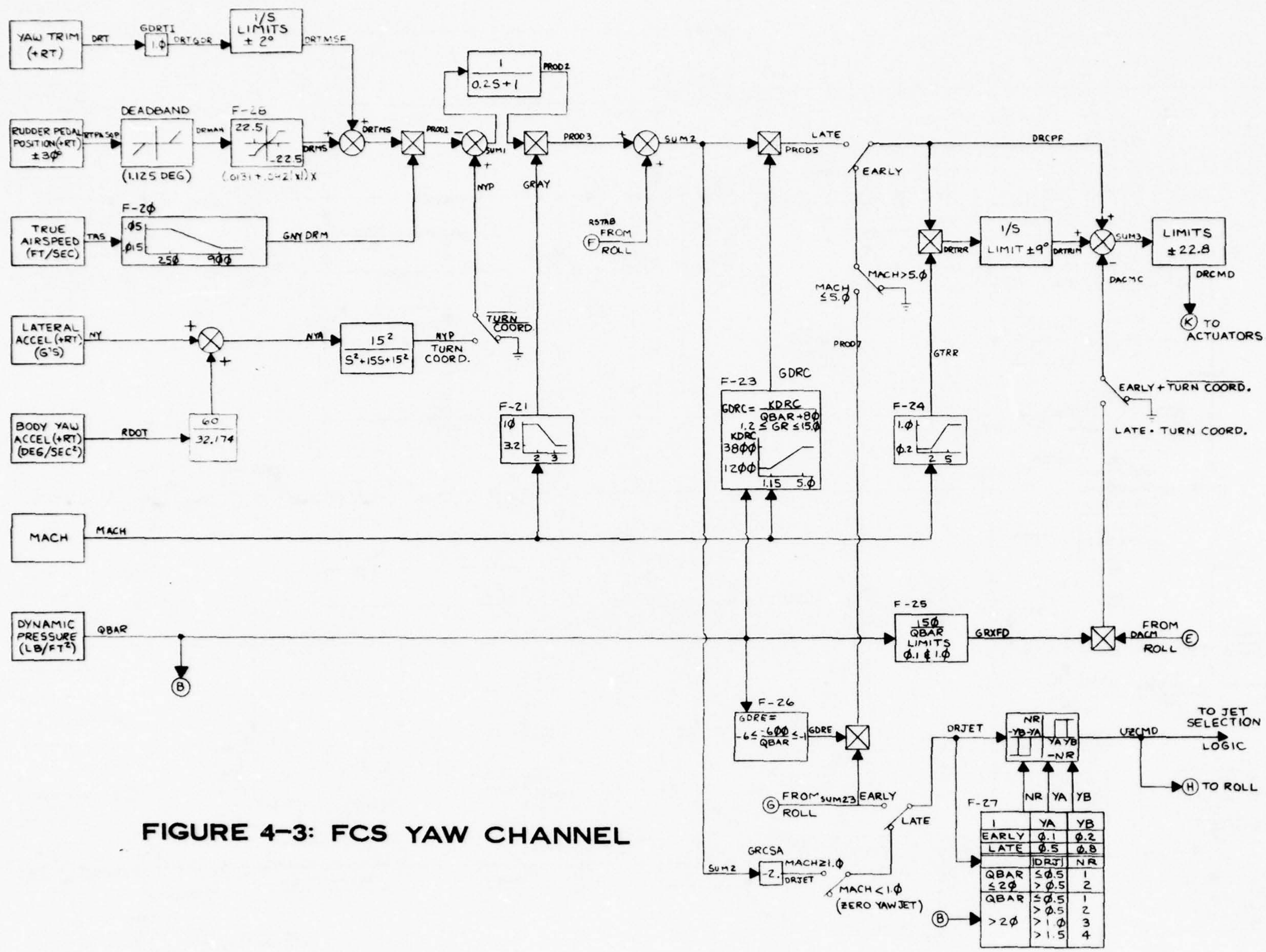
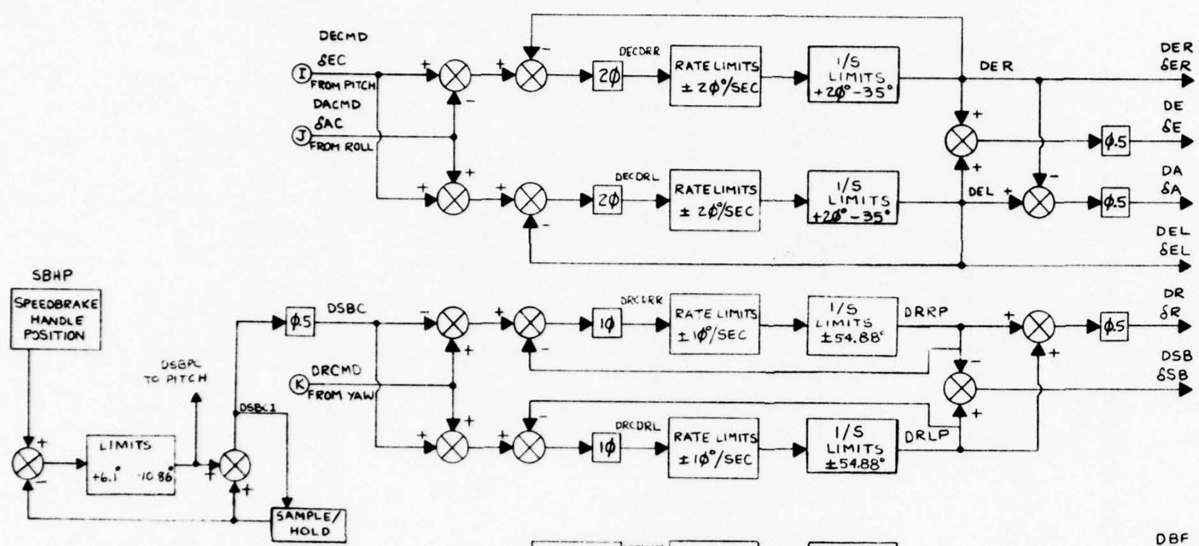


FIGURE 4-3: FCS YAW CHANNEL

FIGURE 4-4: FCS ACTUATORS



differences (backward, forward, or central differences). A more elaborate process is used in this program to prevent computational instability and aliasing problems. It is called Bilinear transformation (simple conformal mapping) and it is represented by the following transformation:

$$s = \frac{2}{T}\frac{(1 - z^{-1})}{(1 + z^{-1})} \tag{4-1}$$

Consider a first-order transfer function as an example. Given

$$H(s) = \frac{10}{s + 10} \tag{4-2}$$

Substituting s for z as dictated by equation (4-1)

$$H(z) = \frac{10}{\frac{2}{T}\frac{(1 - z^{-1})}{(1 + z^{-1})} + 10}$$

$$H(z) = \frac{10T + 10Tz^{-1}}{2 - 2z^{-1} + 10T + 10Tz^{-1}} \tag{4-3}$$

In this program the frame time is T=.04 second. Any first-order transfer function can be, for our programming purposes, reduced to the form shown below:

$$\frac{Y}{X} = \frac{G_1 + G_2 * z^{-1}}{1 + G_3 * z^{-1}} \tag{4-4}$$

Similarly we put all second-order transfer functions in the following form prior to digital coding:

$$\frac{Y}{X} = \frac{G_1 + G_2 * z^{-1} + G_3 * z^{-2}}{1 + G_4 * z^{-1} + G_5 * z^{-2}} \tag{4-5}$$

Thus the z-transform of equation (4-2) finally becomes, after introducing the proper value for T into equation (4-3)

$$H(z) = \frac{.1667 + .1667 z^{-1}}{1 - .6667 z^{-1}} \tag{4-6}$$

Computer mechanization of equations (4-4) and (4-5) are done in the form of function subroutines called FILT1 and FILT2 respectively. FILT1 and FILT2 stand for first-order filter and second-order filter. For FILT1 the expression comes directly from equation (4-4)

$$Y_n = G_1 * X_n + X_{OLD} \tag{4-7}$$

$$X_{OLD} = G_2 * X_{n-1} - G_3 * Y_{n-1}$$

For FILT2, equation (4-5) can be put into the format

$$Y_n = G_1 * X_n + X_{node1} \tag{4-8}$$

where

$$X_{node1} = G_2 * X_{n-1} - G_4 * Y_{n-1} + X_{node2}$$

$$X_{node2} = G_3 * X_{n-2} - G_5 * Y_{n-2}$$

Refer to Table 4-1 to correlate an s-domain transfer function to its z-domain counterpart. Besides the filters, the hysteresis transfer function also deserves a brief mention. Figure 4-5 depicts the flowchart of the HYSTER subroutine and it is self-explanatory.

A detailed flowchart of the Shuttle Flight Control System (SHTLFCS subroutine) is contained in Figures 4-6. The actual listing of SHTLFCS can be found in Appendix D.

TABLE 4-1 : Transfer Functions

Transfer Functions (s-domain)	z-Transform Equivalent (for sampling period T=.04 sec)
$\frac{1}{s}$	$\frac{.04}{1 - z^{-1}}$
$\frac{s}{s+1}$	$\frac{.9804 - .9804 z^{-1}}{1 - .9608 z^{-1}}$
$\frac{5}{s+5}$	$\frac{.0909 + .0909 z^{-1}}{1 - .8182 z^{-1}}$
$\frac{7.5}{5s+1}$	$\frac{.02988 + .02988 z^{-1}}{1 - .992 z^{-1}}$
$\frac{10}{s+10}$	$\frac{.1667 + .1667 z^{-1}}{1 - .6667 z^{-1}}$
$\frac{1}{s+1}$	$\frac{.01961 + .01961 z^{-1}}{1 - .9608 z^{-1}}$
$\frac{20s}{20s+1}$	$\frac{.999 - .999 z^{-1}}{1 - .998 z^{-1}}$
$\frac{s+1}{.5s+1}$	$\frac{1.9615 - 1.8846 z^{-1}}{1 - .9231 z^{-1}}$
$\frac{15^2}{s^2 + 15s + 15^2}$	$\frac{.06475 + .1295 z^{-1} + .06475 z^{-2}}{1 - 1.309 z^{-1} + .5683 z^{-2}}$

Figure 4-5: Hysteresis Function

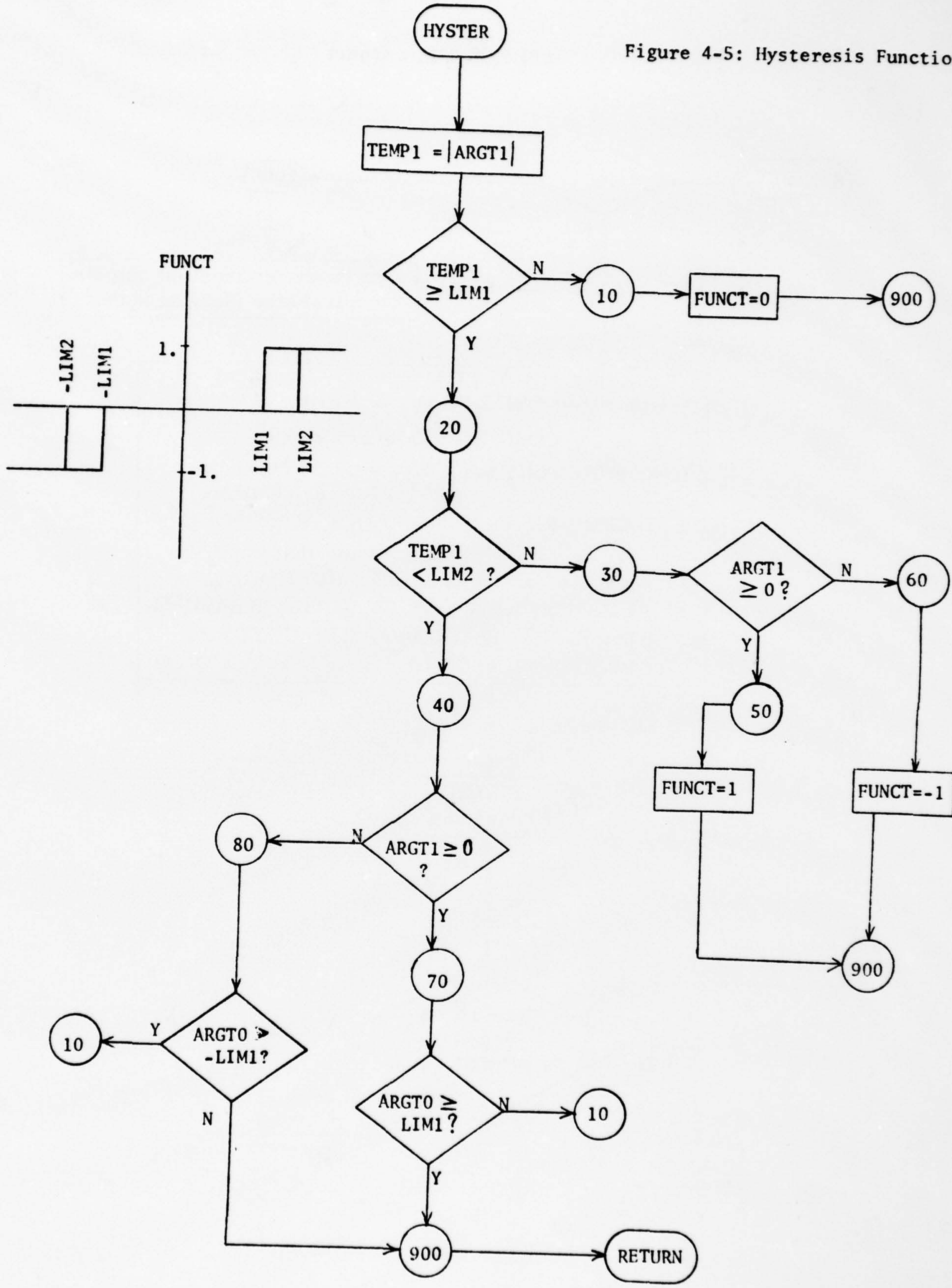


Figure 4-6: SHTLFCS Flow Chart

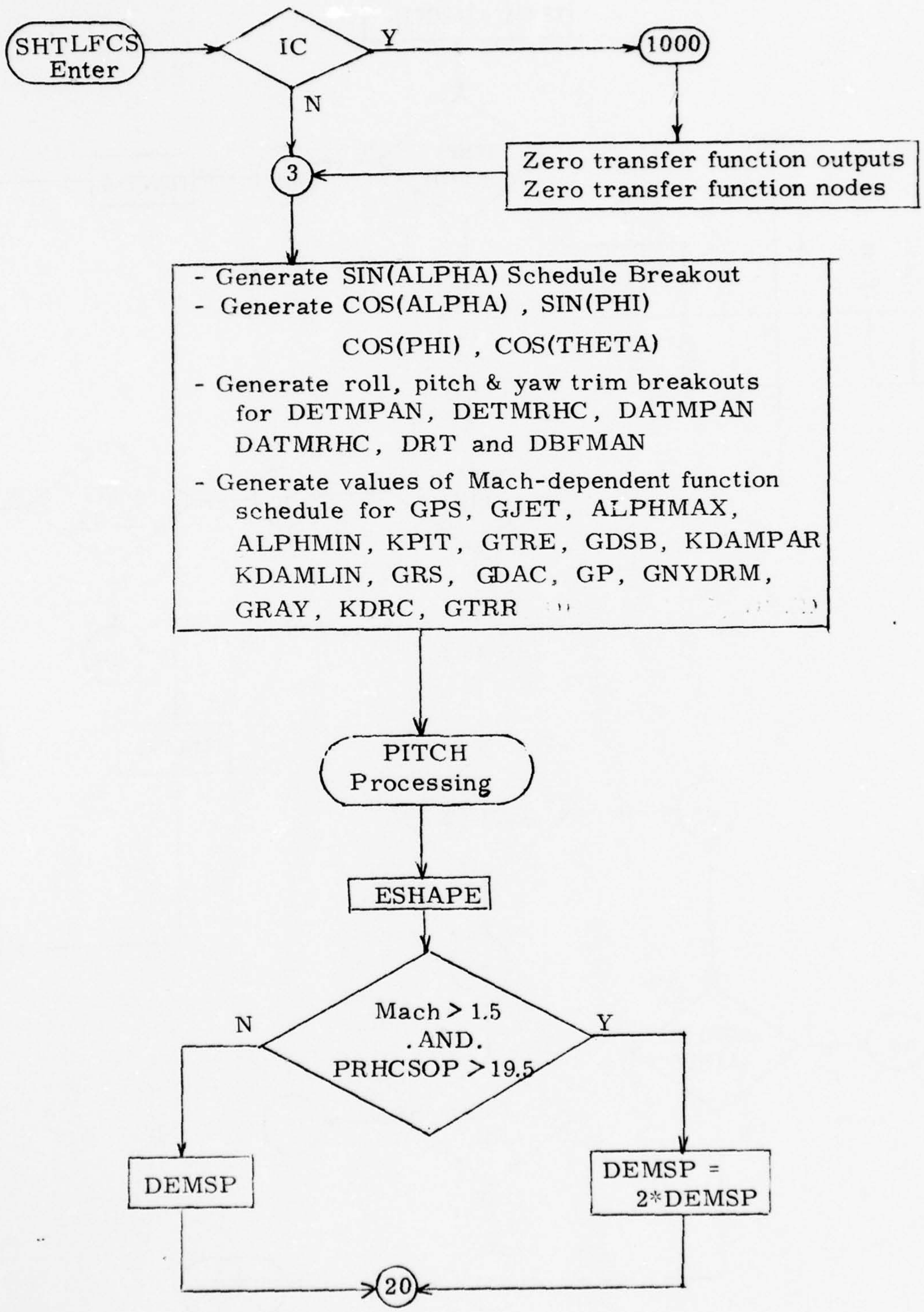
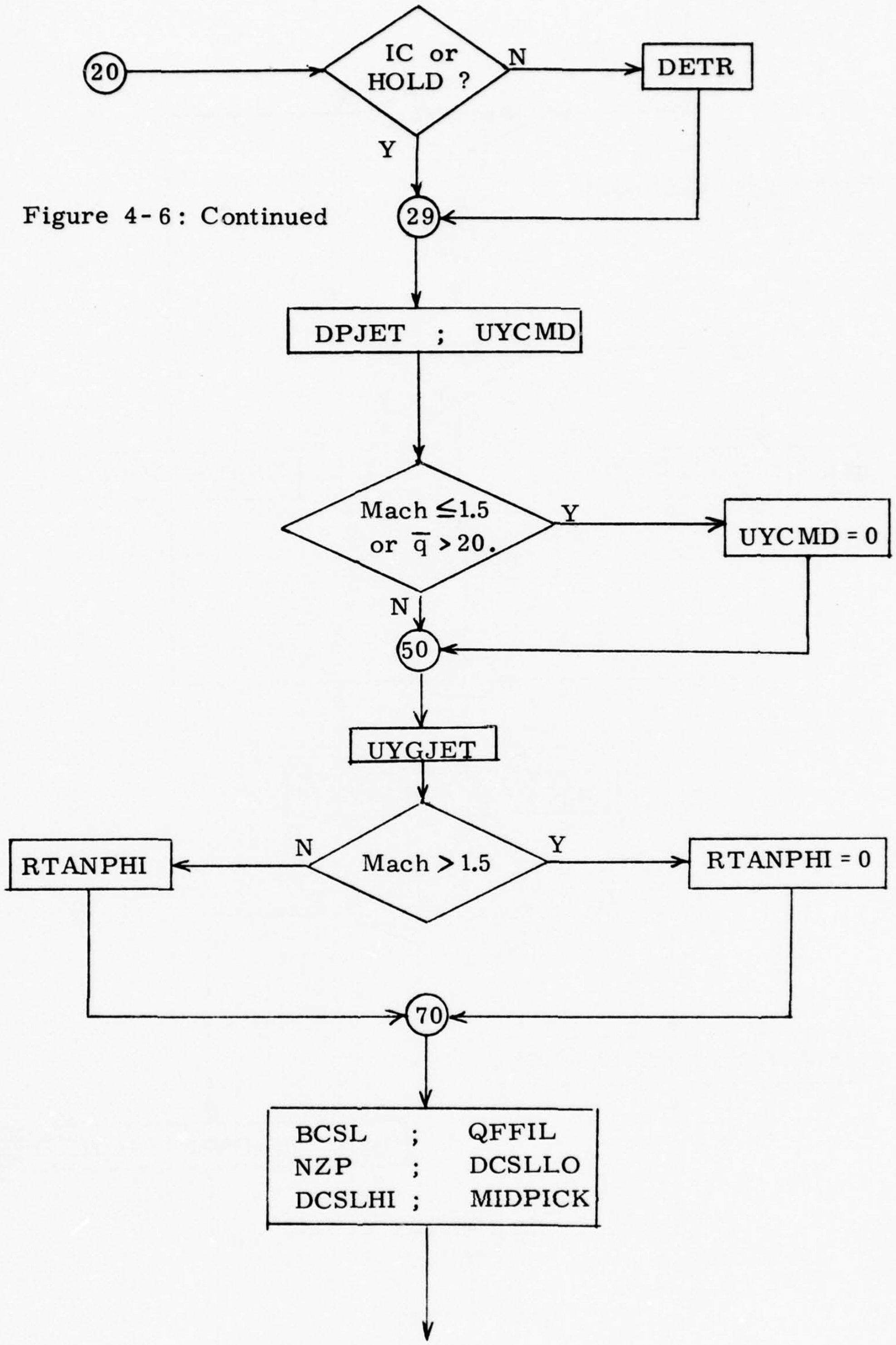
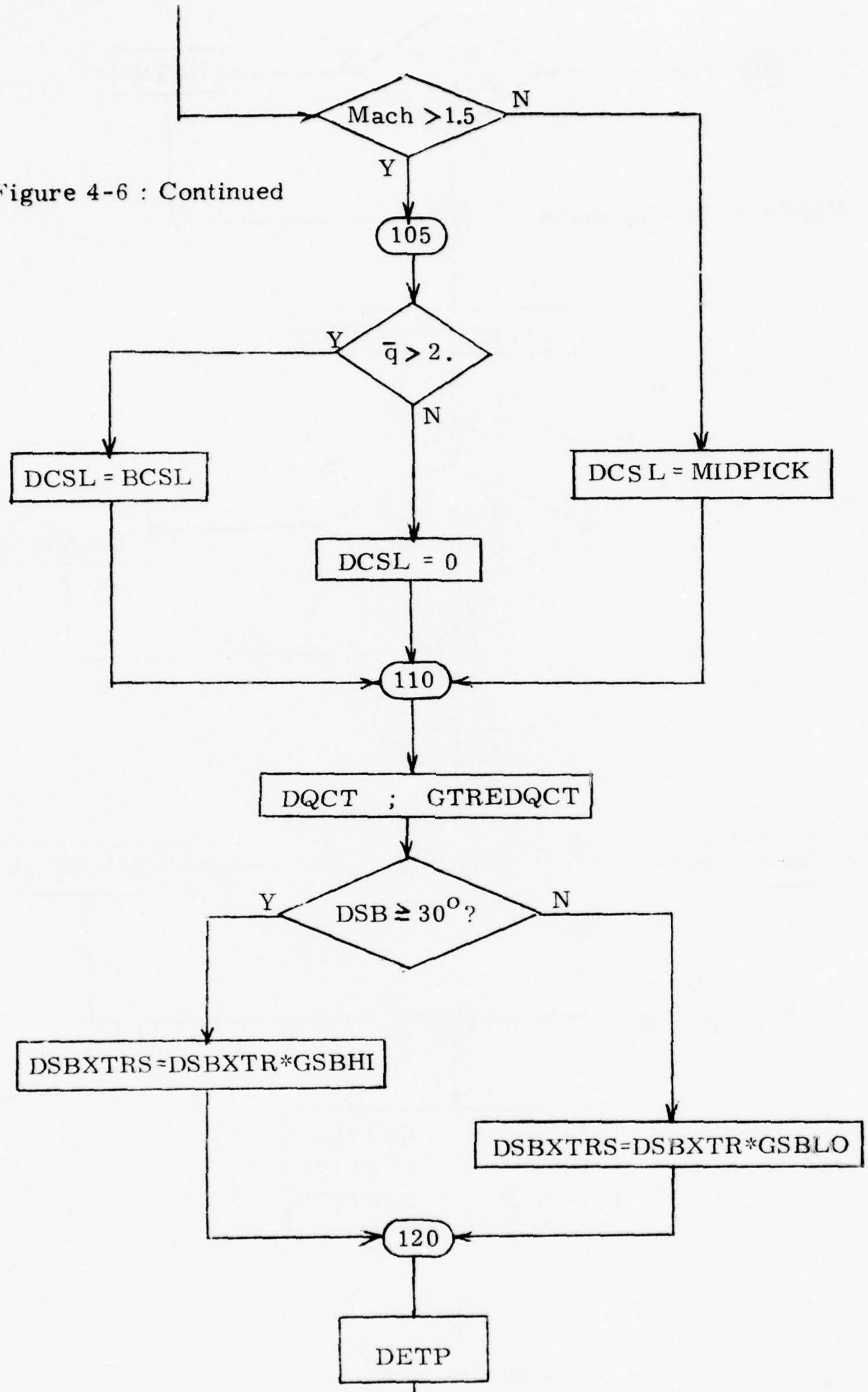


Figure 4-6: Continued



To be continued
on next page

Figure 4-6 : Continued



To be continued on next page

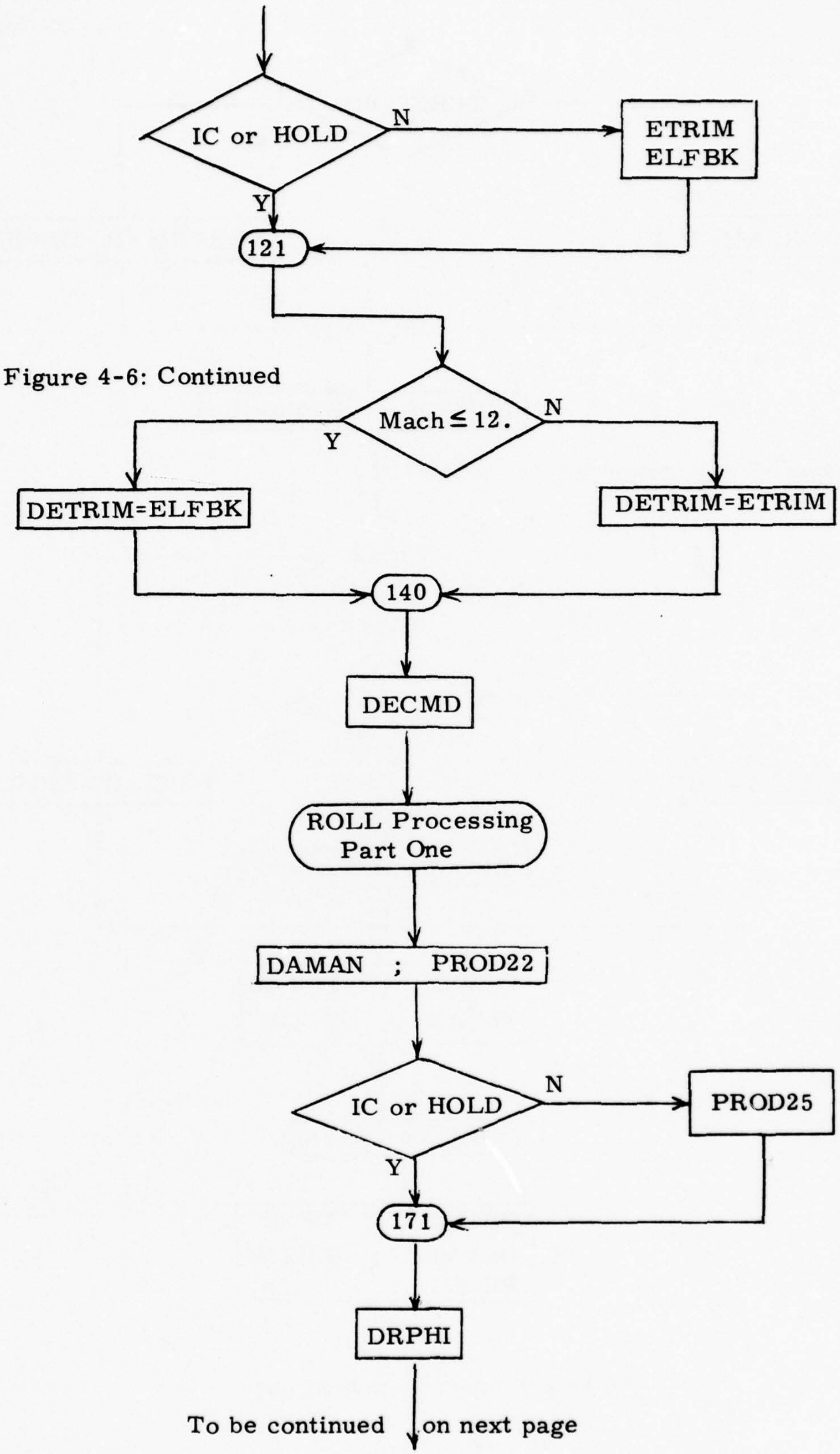


Figure 4-6: Continued

To be continued on next page

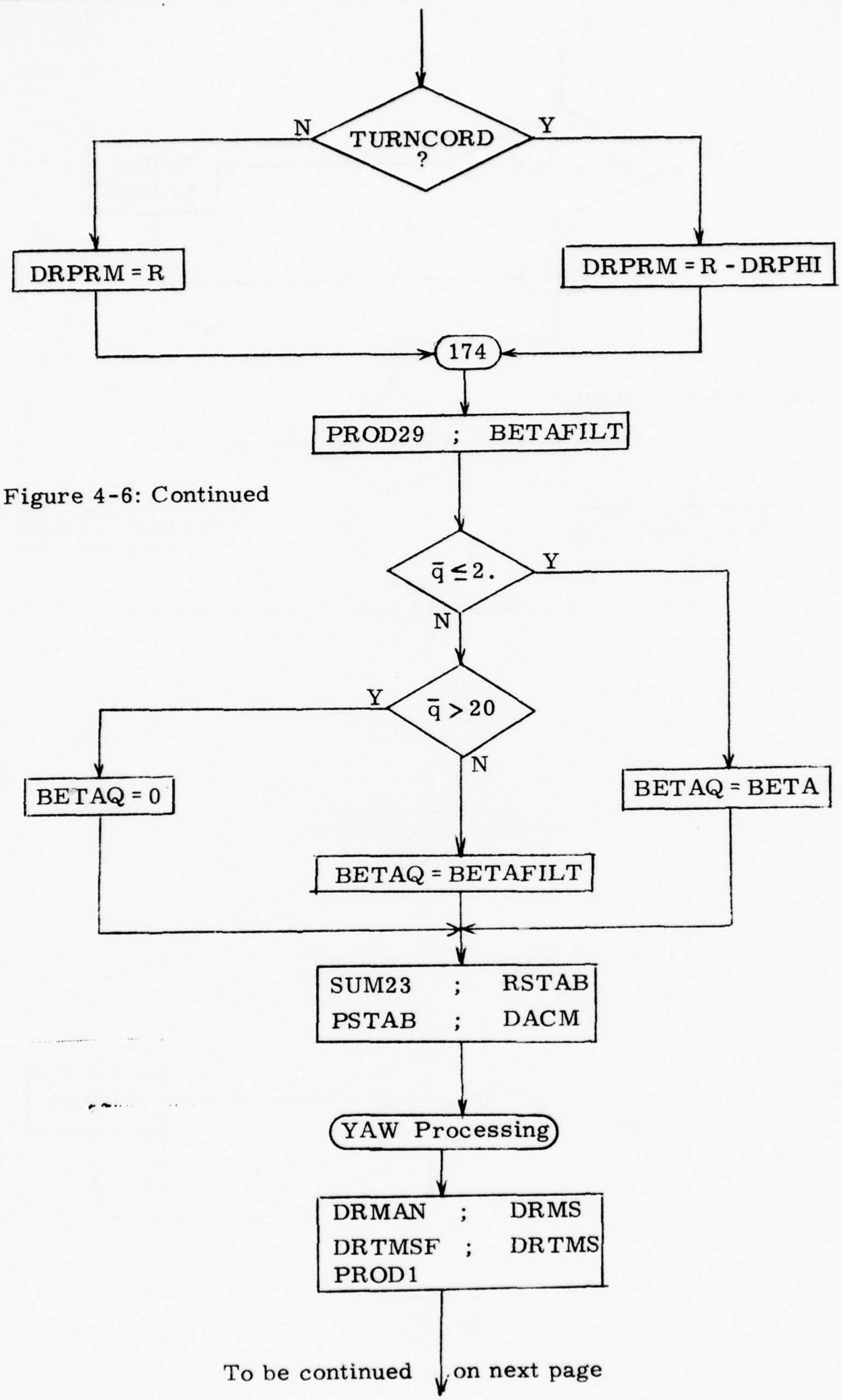


Figure 4-6: Continued

To be continued on next page

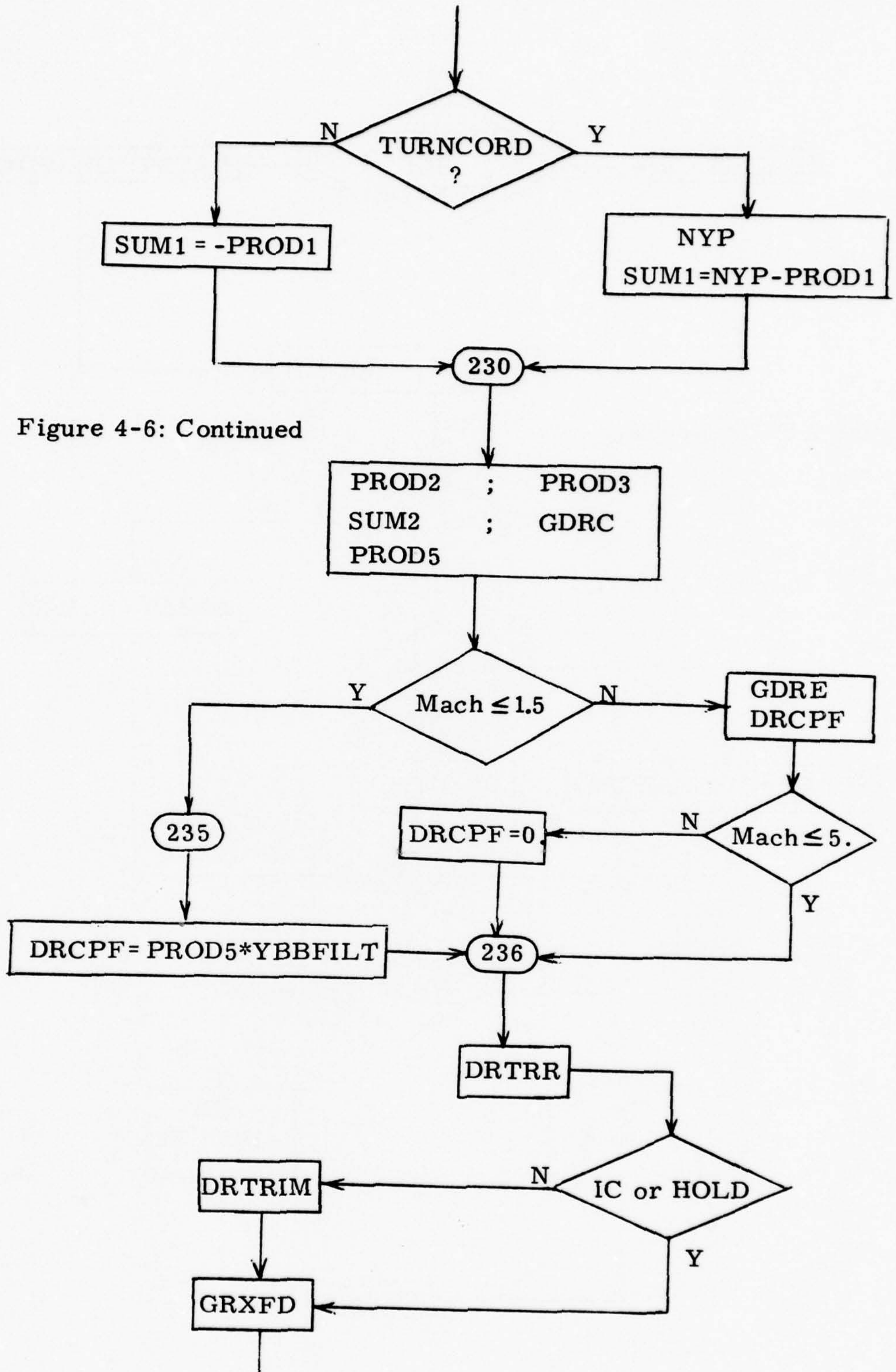


Figure 4-6: Continued

To be continued on the next page

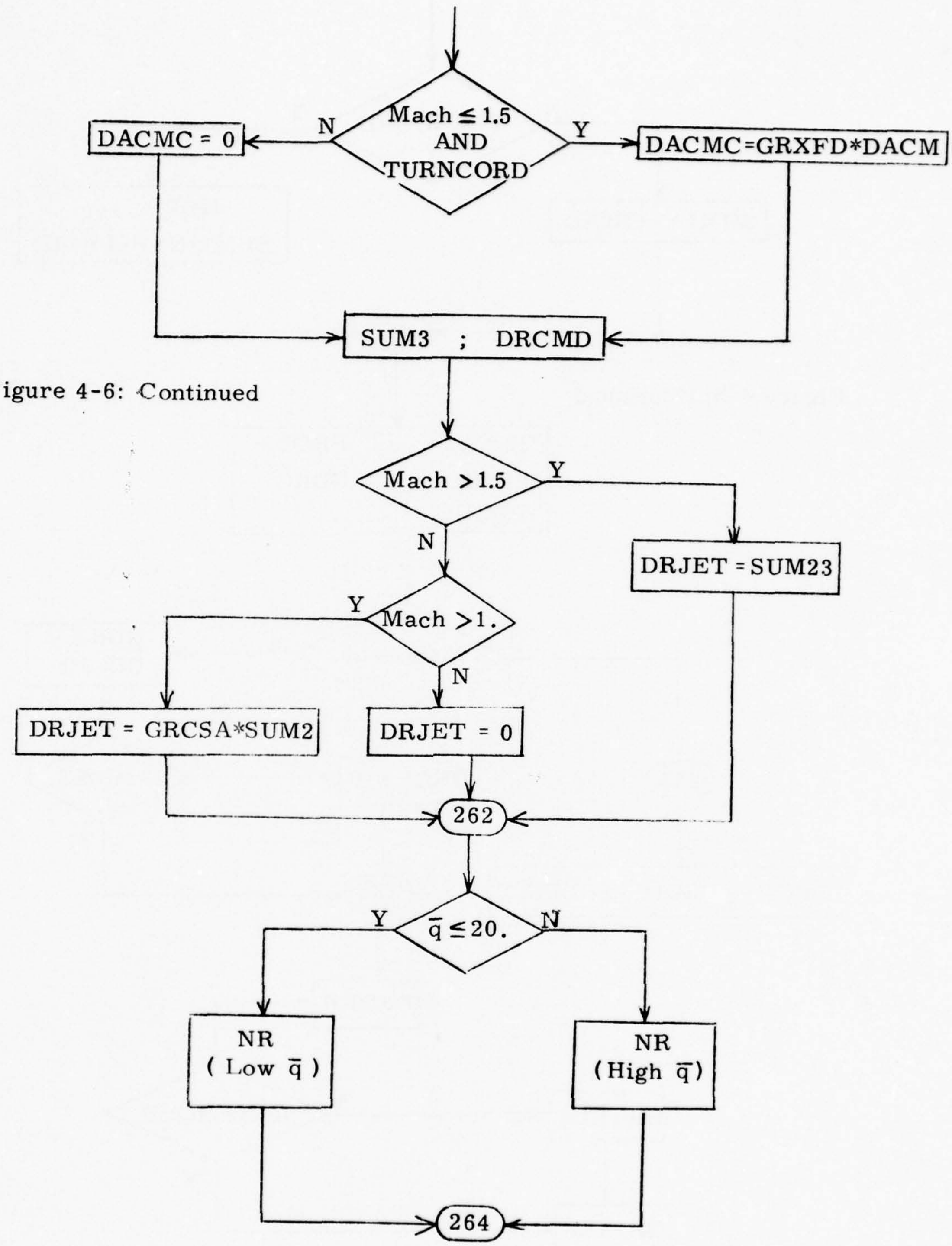


Figure 4-6: Continued

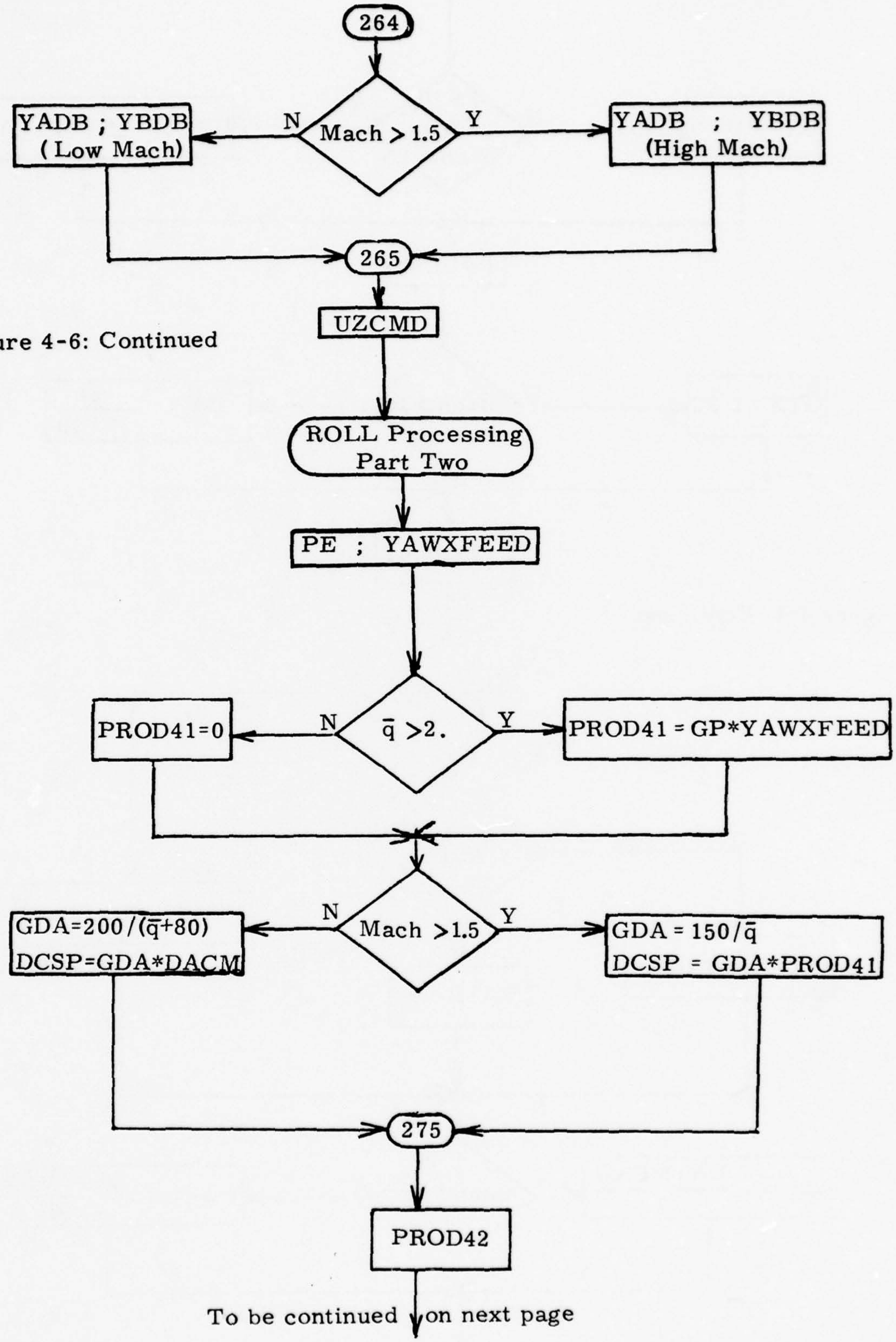


Figure 4-6: Continued

To be continued on next page

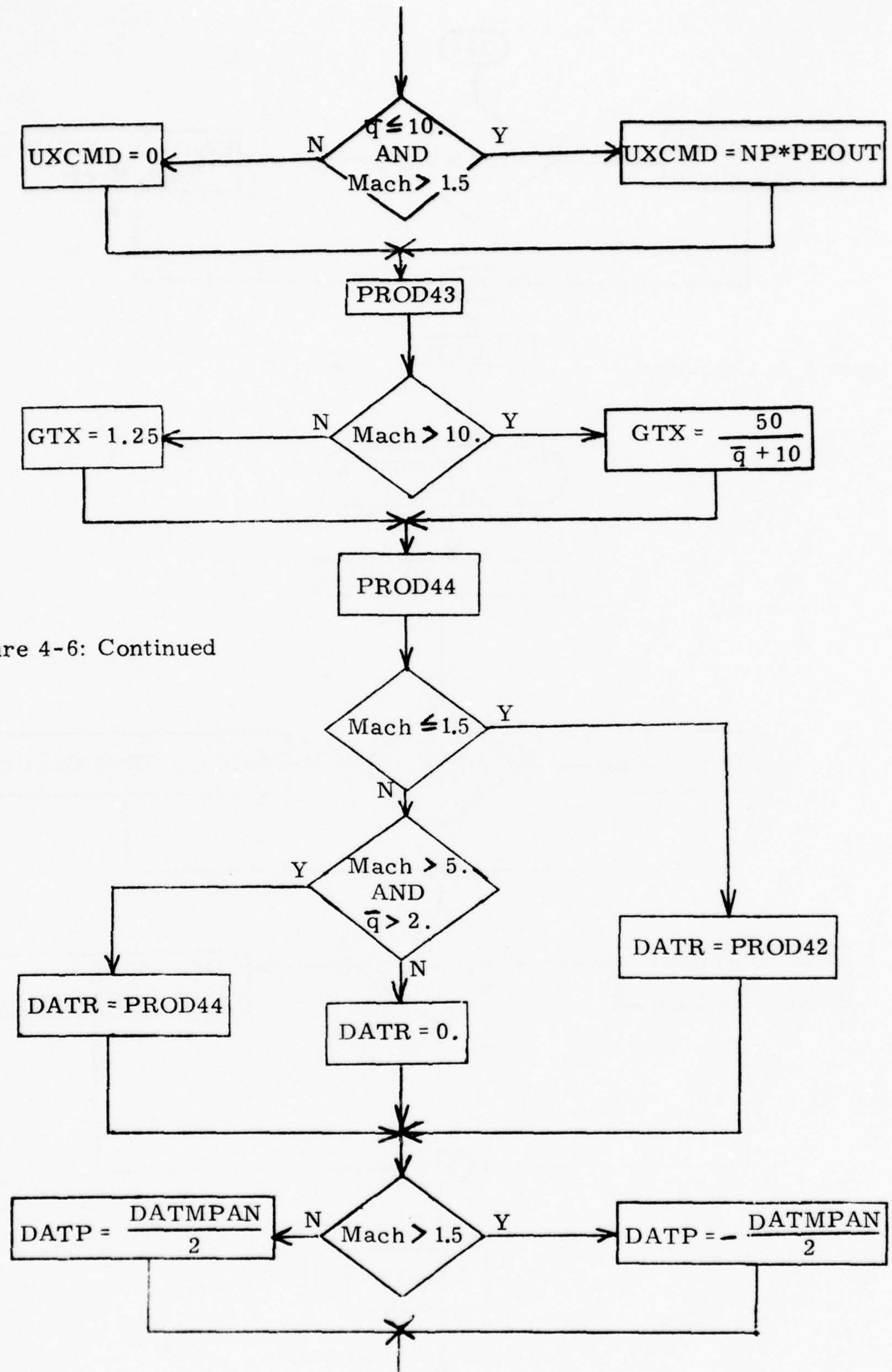


Figure 4-6: Continued

To be continued on next page

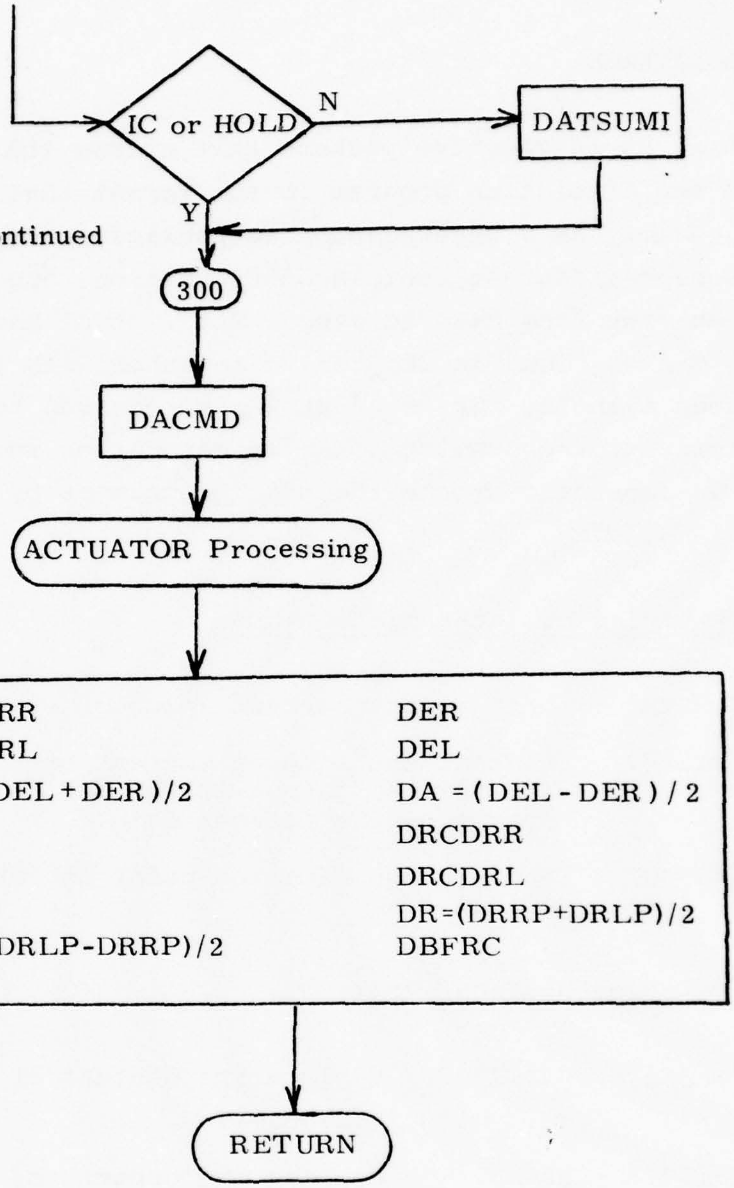


Figure 4-6 : Continued

5. EASE PACKAGE

EASE is an interactive package that allows the researcher easy access to the Simulation program in the format that is natural to him. To qualify as a general-purpose package, the program must readily accept different vehicle configurations and flight conditions which vary from case to case. The list of parameters and variables can be found in Chapter VI together with their units and computer symbols. Users of EASE are required to follow two simple formats corresponding to an interrogation and an input. EASE can be accessed through the Main program or by setting Control Switch 12 to .TRUE. .

a/ Examples of Interrogation Mode

THETA?	What is the current value of θ? (in F25.10 format)
*12AF?	What is the integer content of address 12AF? (In decimal format I25). The asterisk (*) specifies the integer mode.
43600?	What is the floating-point content of address 43600?

b/ Examples of Input Mode

OMEGA = .000072923	Replace the content of OMEGA with .72923E-4.
*17777 = 50	Replace the content of cell 17777 with 50 decimal
1FF2C = .25	Put .25 (in floating-point format) into cell 1FF2C.

c/ To exit EASE, type an exclamation mark "!" followed by a carriage return (C/R).

For unusally large numbers or unusually small numbers, check the

accompanying listing to avoid truncation of the input data. For example, the I/O floating-point number is determined at format F25.10. Any digits beyond 10 places to the right of the decimal point will be ignored. The user should also safeguard against entering a value for a parameter that is larger than the largest allowable for the parameter. For example, the pitch rate Q_B has a range of ± 2 rad/sec and is scaled at $[Q_B/2]$ when appearing to the Flight Control System. If the user of EASE inadvertently sets $Q_B = 3$, the output of the DAC (digital-to-analog converter) corresponding to Q_B is completely wrong.

When a wrong symbol was typed, EASE will respond with "NO SUCH SYMBOL IS DEFINED TO EASE". There are other interactive features built into EASE to help the users in case of mistakes. Here are a few common typing mistakes to serve as examples:

a/ OMEGA = 2.6F19

EASE will come up with the message "MEMORY DATA MUST BE A DECIMAL NUMBER" because there is no provision for EASE to accept hexadecimal numbers.

b/ *RHO? (even though ρ was declared a floating-point in the main program). The resulting message is "WRONG INPUT FORMAT".

c/ BYTE = 377 (even though BYTE was declared as 8 bits) In that case $255 \leq \text{BYTE} \leq 0$. EASE will alert the user with the warning "DATA OUTSIDE RANGE".

The simplified flowchart for the EASE package is contained in figure 5 -1 for reference purposes.

Figure 5-1: EASE Flowchart, Executive

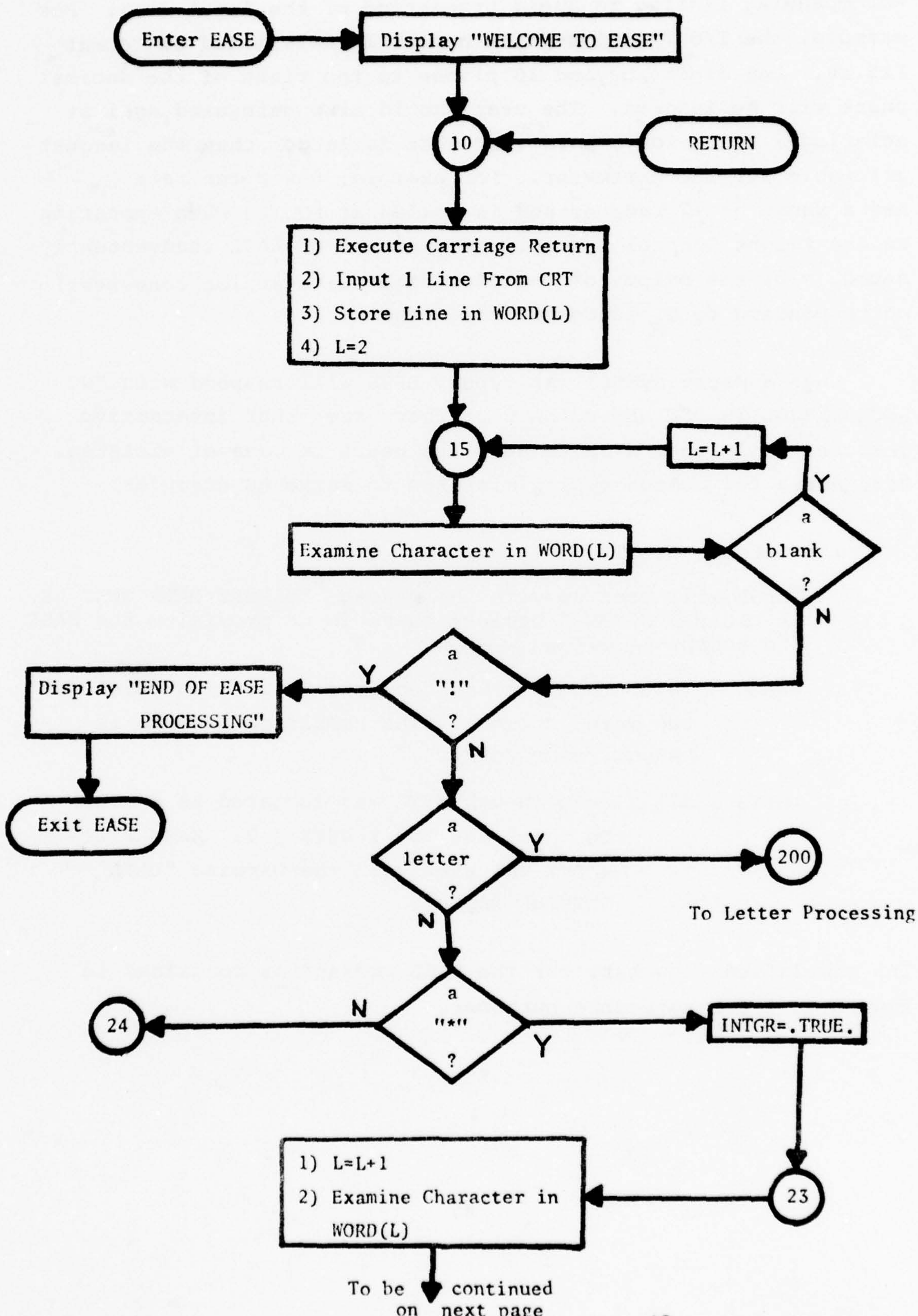


Figure 5-1 (continued), Executive

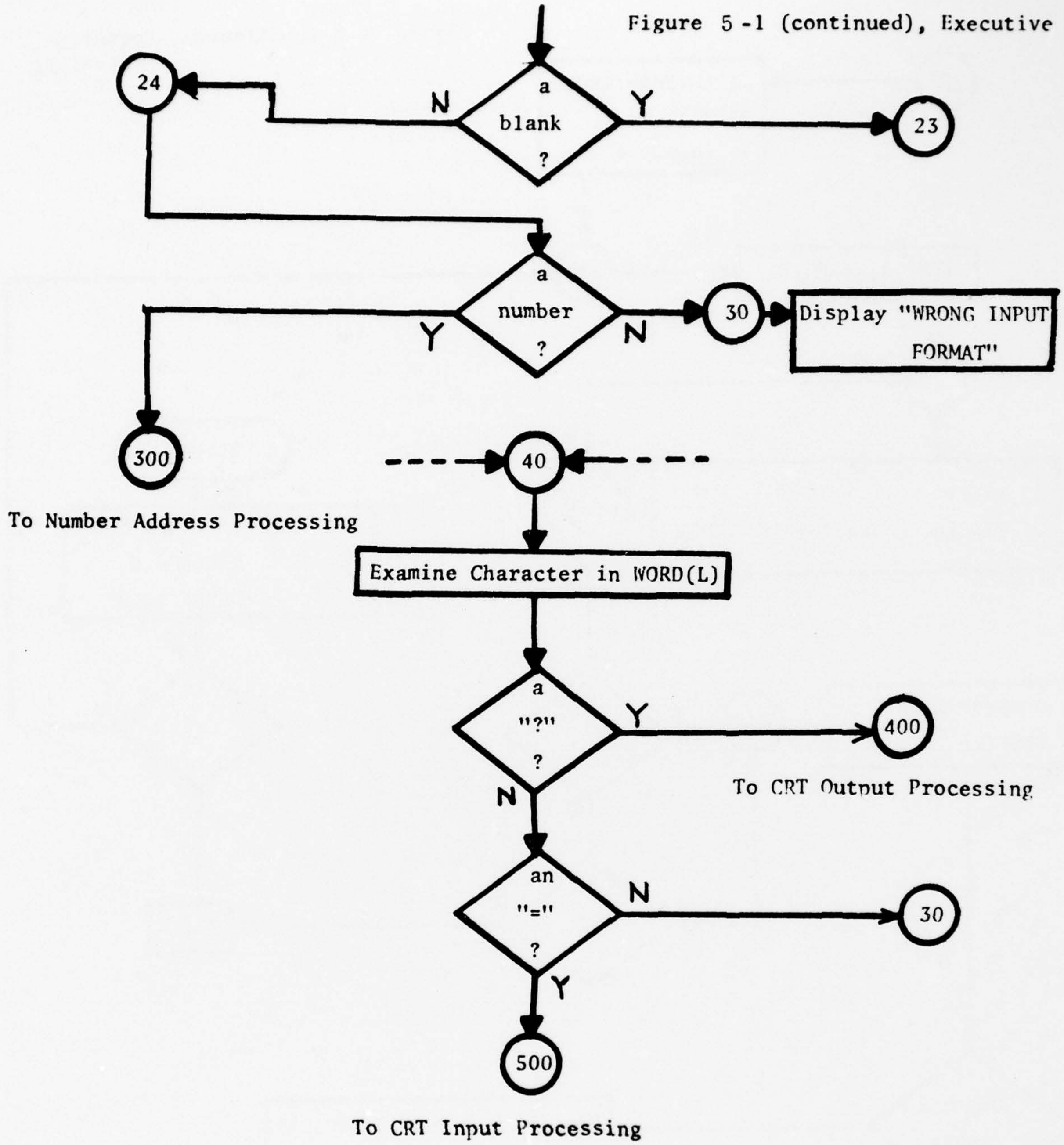


Figure 5-1 (continued), Letter Processing

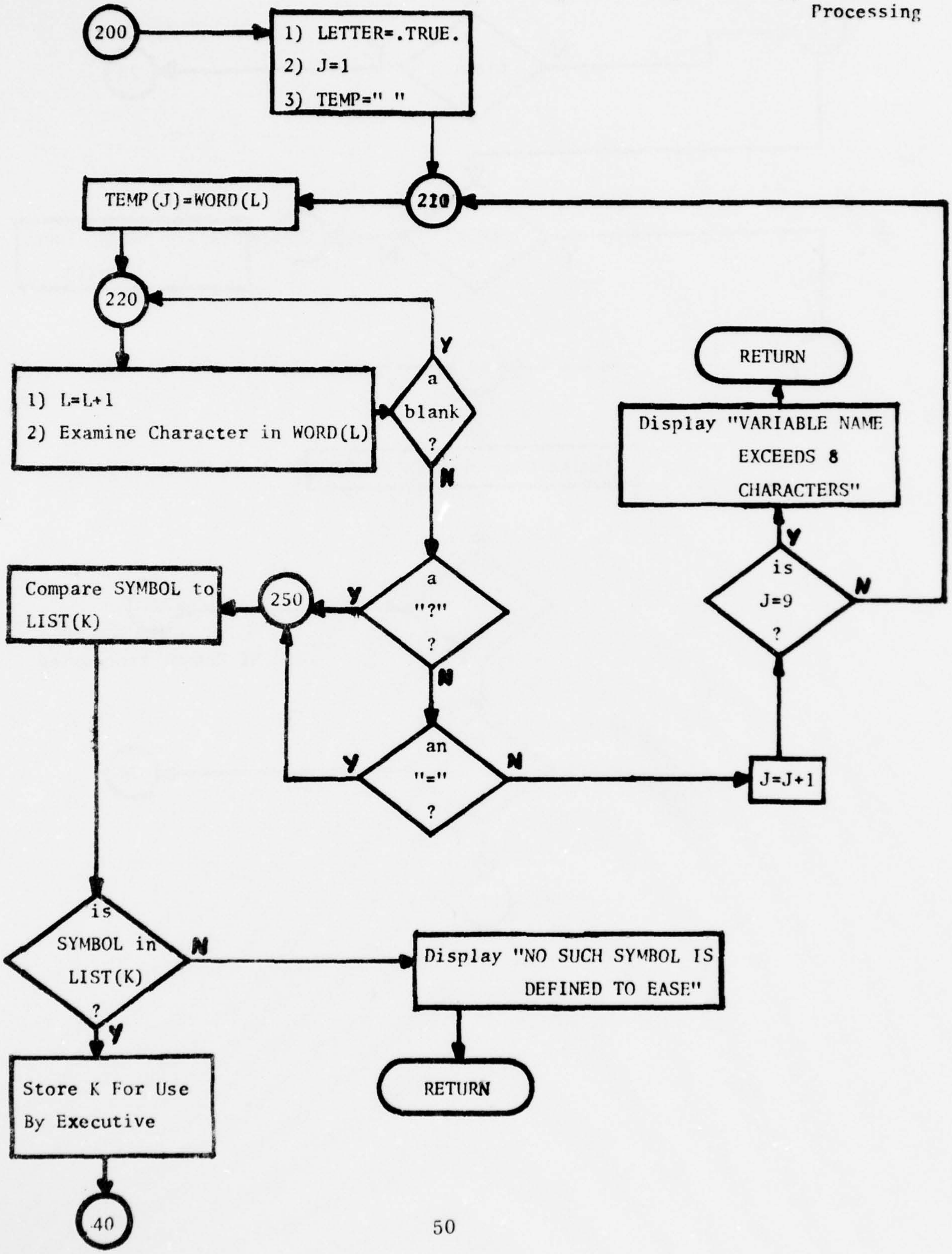


Figure 5 -1 (continued), Number Address Processing

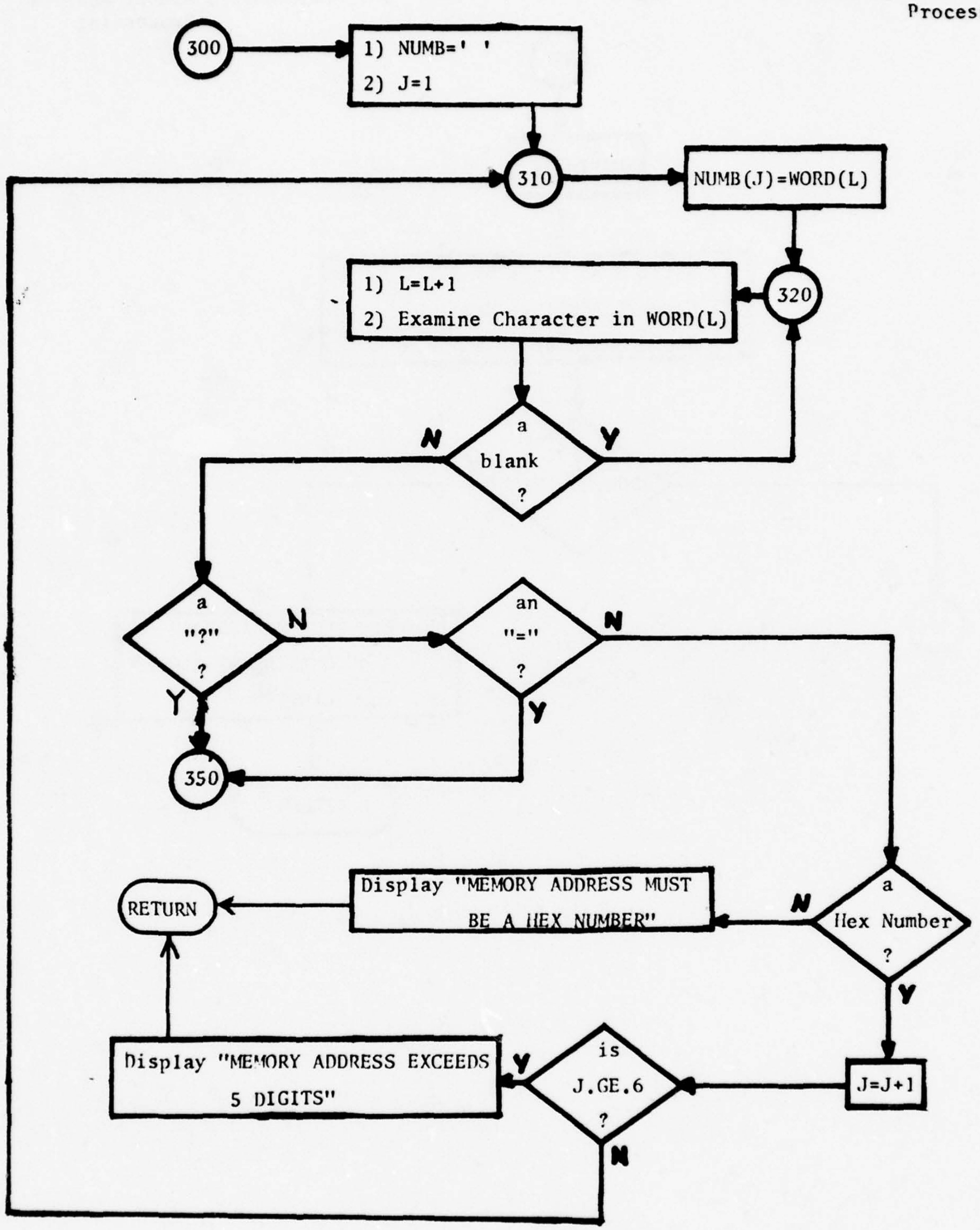


Figure 5-1 (continued), Number Address Processing

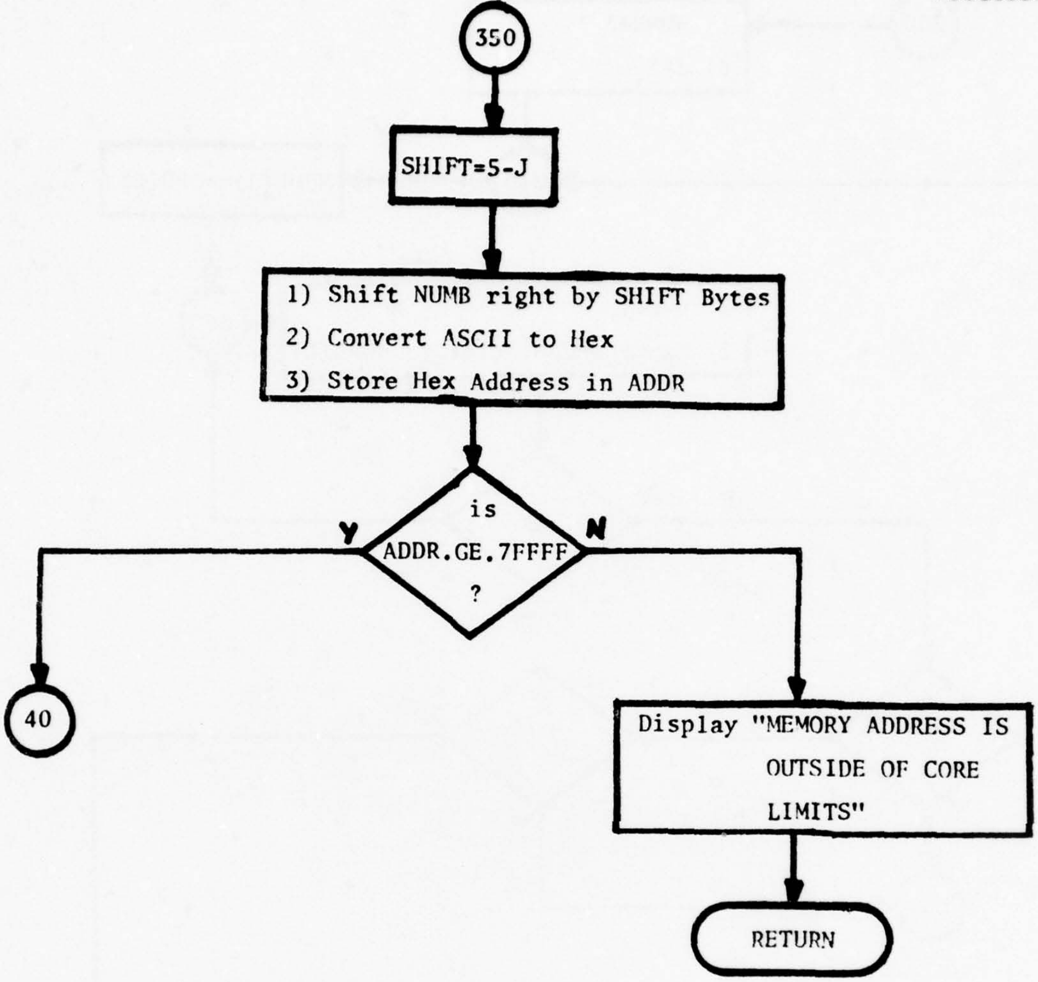


Figure 5-1 (continued), CRT Output Processing

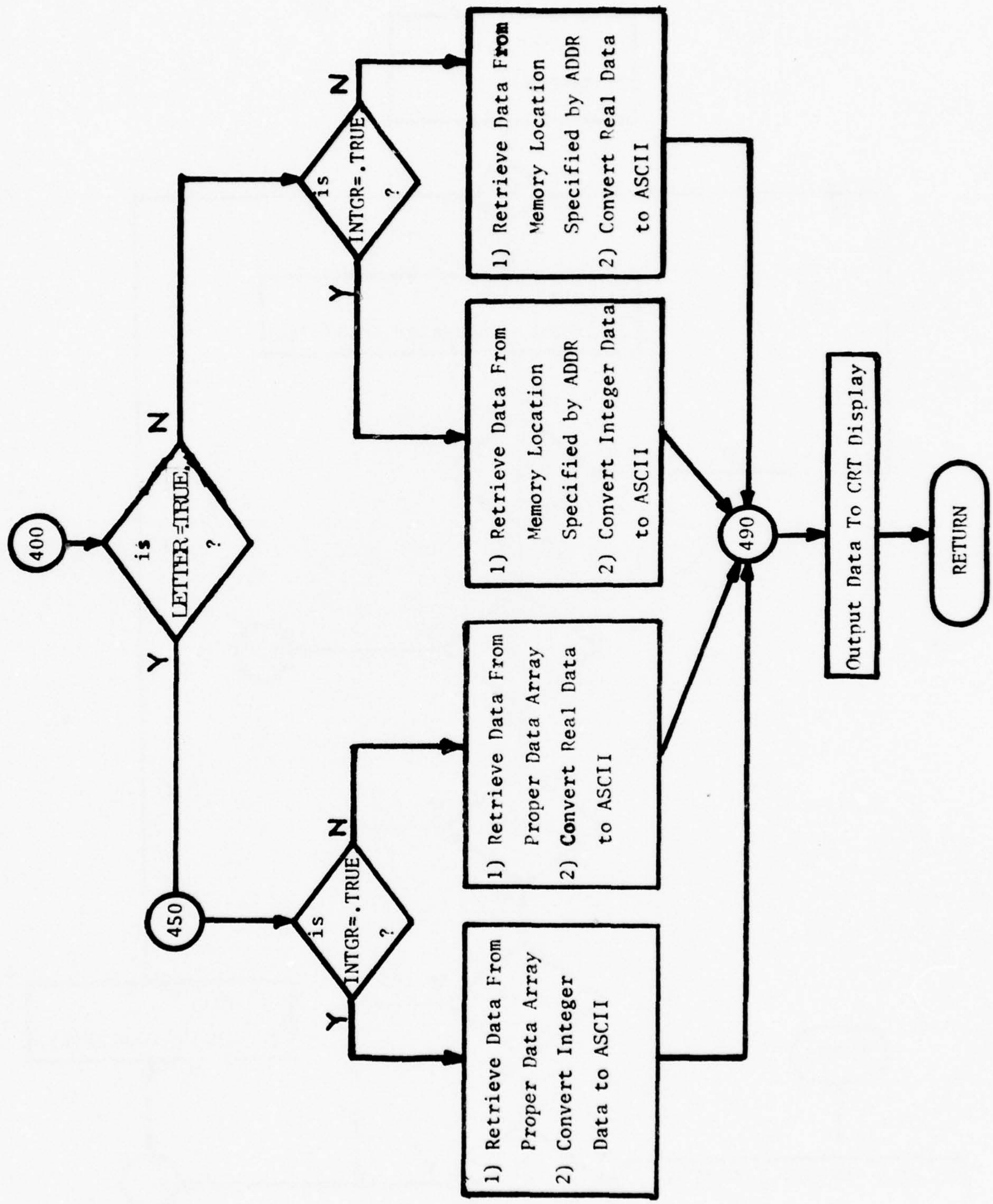


Figure 5-1 (continued), CRT Input Processing

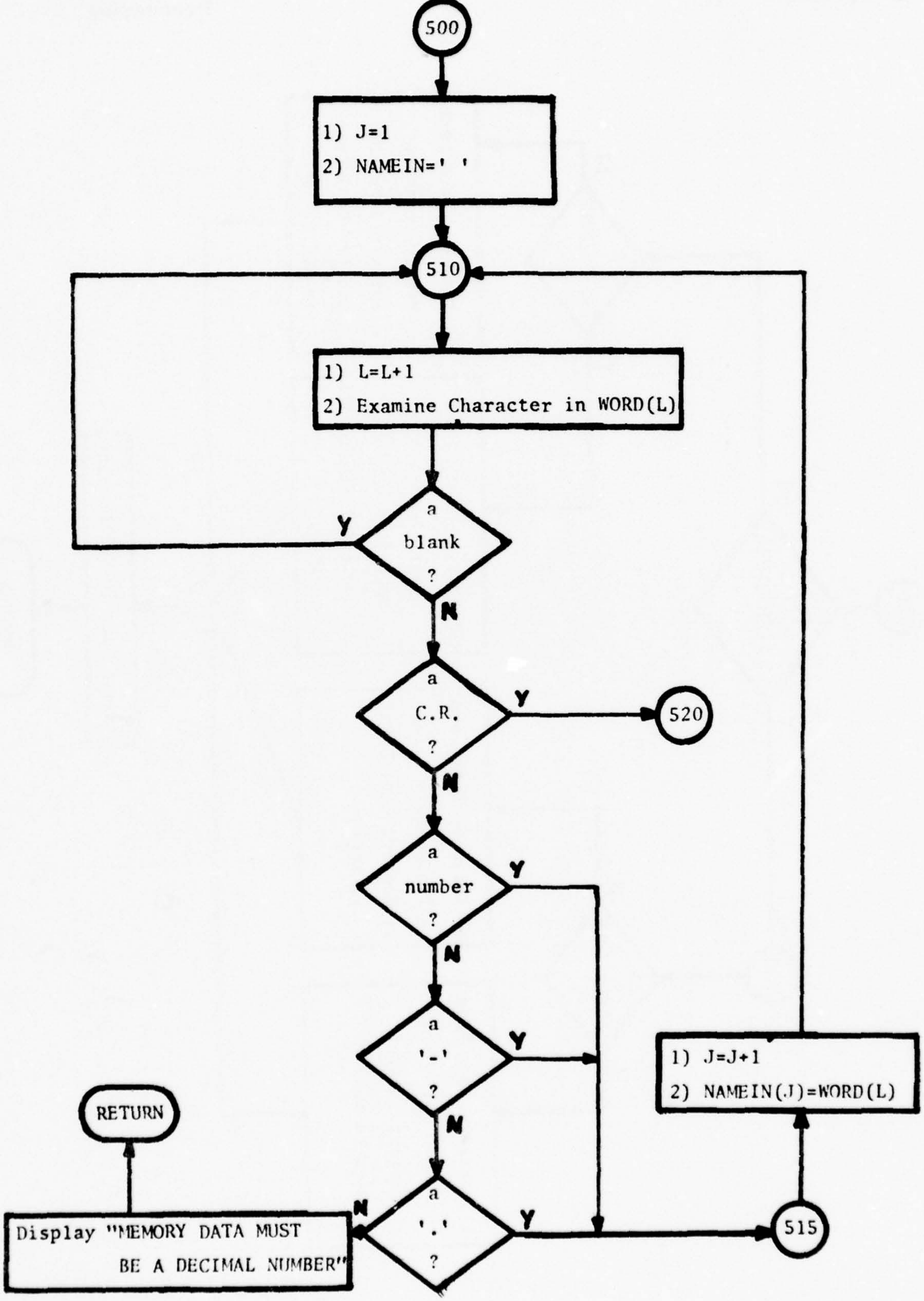
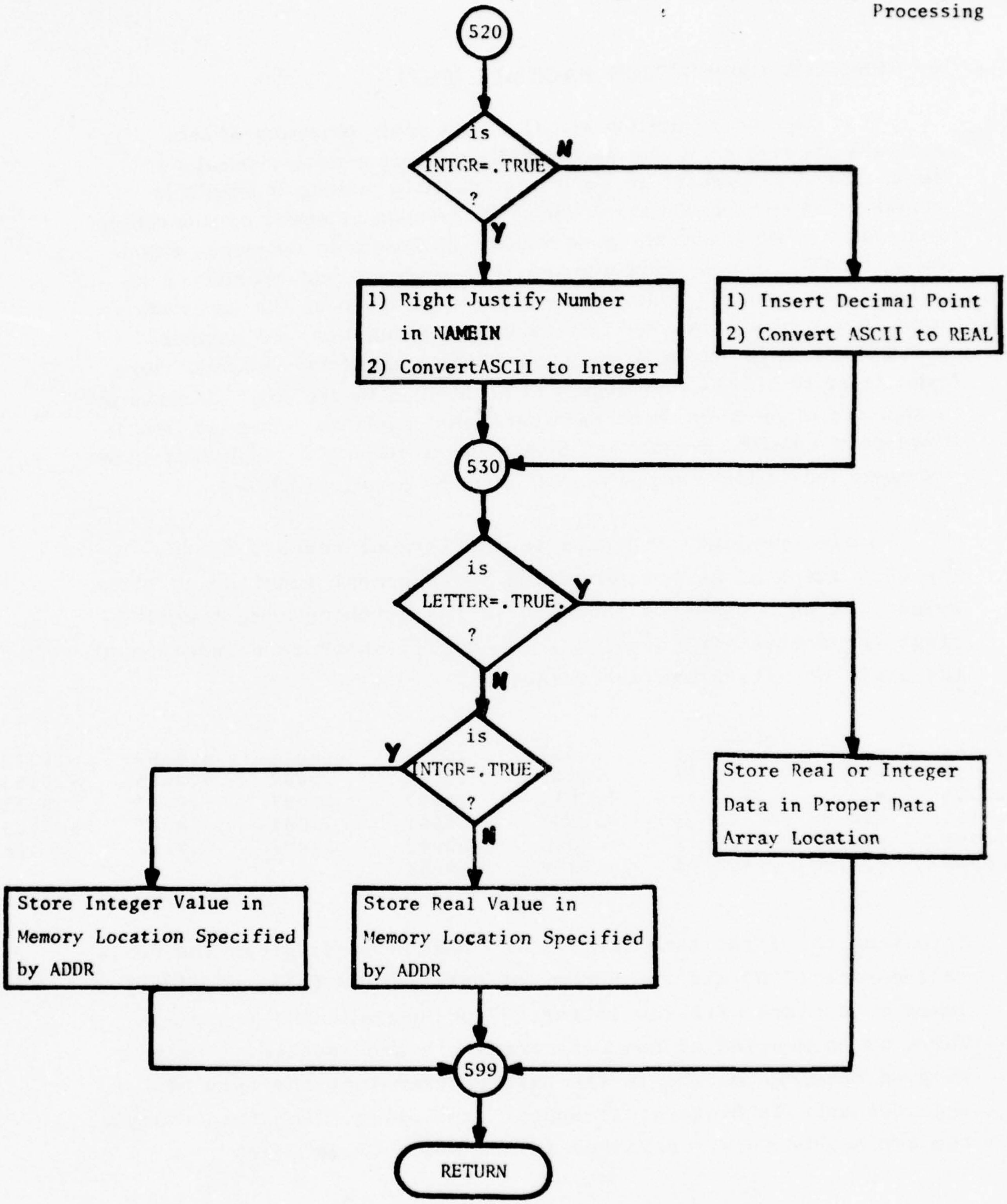


Figure 5-1 (continued), CRT Input Processing



6. FUNCTION GENERATION PACKAGE (FGP)

Fast function generation constitute the main objective of this software package so it can successfully support a time-critical simulation. The package is Fortran compatible making it adaptable elsewhere if needed. On the other hand, because of speed consideration, some part of this program were written in Assemble language. Physically the FGP is embedded into the Main program and considered an integral part thereof. This leature also helps speed up the execution time. Aerodynamic data for this particular simulation are composed of about 40 functions having from one to three arguments. Overall, there exist more than 4000 data points to be handled by the FGP. If there is a shortage of memory space each data point could be converted into a fixed-point quantity, normalized then made to fit into a 16-bit half-word location. But in that case, the FGP must be greatly modified.

We are provided with data in the form of punched cards. The card image of a representative two-argument function is shown below. In this case the function is the pitching-moment coefficient (or derivative) bias, CM_0 (α, Mach), which is a function of the angle of attack and the velocity.

CMO	45						
-.0083	-.0023	0.0047	.0057	.0037	-.0013	-.0153	-.0373
-.0553	-.0083	-.0023	.0047	.0057	.0037	-.0013	-.0153
-.0303	-.0503	-.0073	-.0013	.0057	.0057	.0027	-.0033
-.0165	-.0283	-.0413	-.0043	.0007	.0081	.0047	-.0003
-.0093	-.0193	-.0273	-.0263	.0002	.0052	.0132	.0072
-.0038	-.0158	-.0248	-.0278	.0052			

Note that the first card (called the Header card) gives the derivative name (CMO) and the number of data points (45). Derivative names must start with the letter "C" or they will be rejected . There is no mention of how many arguments are involved. Only when we make use of CM_0 in the Main program that the type of argument and its breakpoints appear explicitly. For this example, the two arguments are arranged in the order $CM_0(\alpha_1, M_1)$;

$CM_0(\alpha_2, M_1)$; $CM_0(\alpha_3, M_1)$; . . . . .; $CM_0(\alpha_5, M_1)$; $CM_0(\alpha_1, M_2)$;
$CM_0(\alpha_2, M_2)$; $CM_0(\alpha_3, M_2)$; . . . . . until $CM_0(\alpha_5, M_9)$.

The FGP is made up of three separate sub-programs, each doing a specified task. They are: DATASTORE, POINT and DERIVE4. Their particular roles are described below.

6.1 DATASTORE Program : This program performs the following tasks:

- Read aerodynamic data from the card file as described earlier.
- Check the validity of data including the name.
- Rearrange the data in a predetermined order.
- When an end-of-file (EOF) card is encountered, write the whole data array on a disc file.

The flowchart for DATASTORE can be found on figure 6-1. We can analyze DATASTORE program by tracing through the handling of one stability derivative, say CM0 again. When MAIN program calls DATASTORE, it reads and inspects the first card, also called the Header card. That card should contain an 8-character derivative name followed by a blank and followed by a 6-digit integers (blanks ƀ included). For example

$$\overbrace{\text{CM0ƀƀƀƀƀ}}^{\text{Derivative name}}\text{ƀ}\underbrace{\text{ƀƀƀƀ45}}_{\text{Number of data points}}\overbrace{\text{COMMENTS . . . .}}^{\text{Columns \#16 thru \#80 reserved for comments}}$$

One header card is required per derivative by DATASTORE. Anytime an EOF card is encountered in place of a header card, control is reverted to the MAIN program. Thus the very last card must be an EOF or the program will hang up.

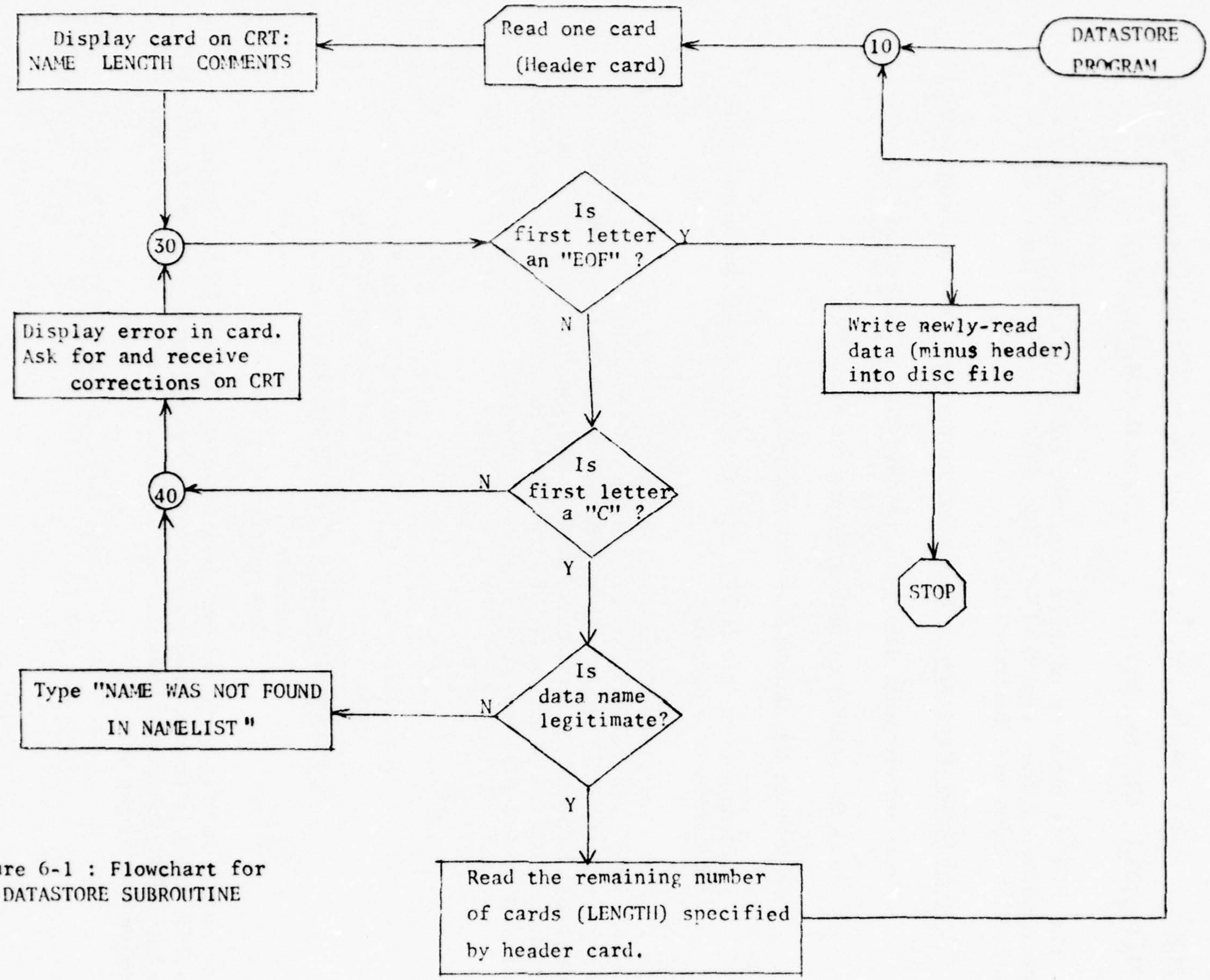


Figure 6-1 : Flowchart for DATASTORE SUBROUTINE

Since all derivative names stored start with the letter "C", a test is made of the first character to eliminate any unnecessary search of the Data List. Only those names previously defined in the Main program's Data Table (such as CM_0, C_{ℓ_β}, $C_{n_{\delta a}}$, ...) are considered legitimate. If the first letter of the header card is a " C ", control is passed to the "name list search" portion of DATASTORE program (beginning with statement #50). In the event the first character of the header card is not a " C ", DATASTORE allows the user the choice of aborting the run (returning to Main program) or the choice of correcting the typographical mistake by inputting a new name on the CRT.

After the derivative name and address have been found in the tables, DATASTORE returns to the card reader and reads the information on the cards stacked behind the header card. The newly arrived data points are stored in the temporary array called DATA(I), (I = 1, LENGTH). In our example LENGTH = 45 as indicated on the header card earlier. When all the expected data points are in (an EOF card is read) DATASTORE terminates the reading phase and all the aerodynamic data are written out on a disc file. Notice that DATASTORE is excuted only once, during program initialization phase and the disc file will be utilized by the Derivative subroutines later on.

It should be aware that DATASTORE is a separate program that must be executed prior to all other programs in the normal sequence of operation. Also note that DATASTORE program must be modified and rerun for any change in the length or table arrangement of the aerodynamic data. These parameters must match those of the MAIN program to insure that the data are stored in the same relative location where they will be used. Specifically care and attention should be paid to the following:

- The order and dimension of the data-table declaration statements
- The DATA statements for variable SIZE and NUMBER

- The ASCII variables in NAMELIST
- The order of the data table ACW (Address Constant Word) array. These parameters effect the proper ordering of the data table in the array which is dumped to disc.

6.2 <u>POINT Subroutine</u> : This subroutine, which is called by the Main program, performs all the preliminary tasks prior to final interpolation. One call to POINT subroutine is required per argument list and a representative calling sequence applicable to our present example might appear as follows:

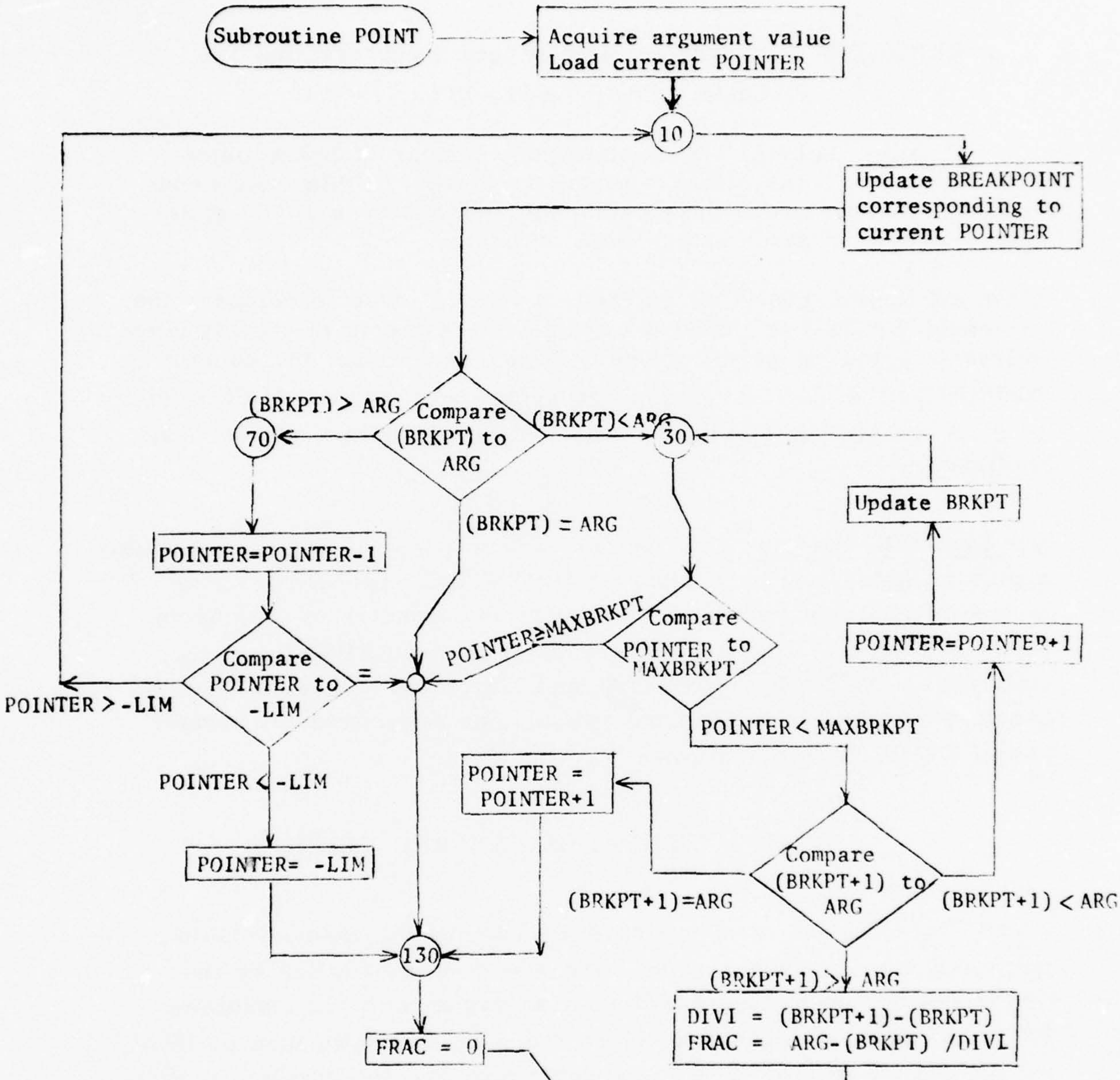
```
                ↑             ↑             ↓             ↓             ↑
CALL POINT(ALPHA, ALPHAPT(2), ALPHAPT(1), ALPHAT(1), ALPHAT(3))

CALL POINT(MACH, MACHPT(2), ALPHAPT(1), ALPHAT(1), ALPHAT(3))
             ↓          ↓            ↑            ↑            ↓
```

The outgoing arrows are added to visually indicate arguments needed by the subroutine for processing the calls and, conversely, the incoming arrows show that information is being returned to the Main program. Two calls are made to POINT subroutine because in our current example , CM0 is a function of two arguments, α and Mach. Let us inspect one of the calls above.

ALPHA = an argument, the present value of α

ALPHAT(3) = starting point of a list of breakpoints associated with the independent variable α. This list consist of floating-point values.

ALPHAPT(1) = returns to Main program as an integer value (called "pointer") indicating the smallest breakpoint adjacent to the argument α, say α_i . This information will be used during subsequent interpolation.



<u>Notes:</u> Abbreviations used:

BRKPT = Breakpoint number

-LIM = 1, the first brkpt

MAXBRKPT= no. of brkpts of
an argument.

ARG=value of the argument

(BRKPT)=the value of the argument
corresponding to BRKPT

Figure 6-2 : Subroutine POINT

ALPHAT(1) = returns to Main program representing the distance, FRAC = $(\alpha - \alpha_i)/(\alpha_{i+1} - \alpha_i)$

Finally, ALPHAPT(2) contains the number of breakpoints associated with the argument. In this case there are 9 breakpoints as declared by a DATA statement in the Main program.

When subroutine POINT is entered, a test is made to compare the argument (α in this instance) against the extrema of the argument values (α_1 and α_9 here). Should $\alpha < \alpha_1$ then we set the pointer ALPHAPT(1) to 1. Conversely whenever $\alpha > \alpha_9$ we set ALPHAPT(1) to 9. A simplified flowchart of the POINT subroutine can be found in figure 6-2.

6.3 <u>Derive Subroutine</u>: This section was written using the FUNCTION approach instead of the standard SUBROUTINE approach in order to reduce program length and minimize the transfer of arguments between different programs. There exist four DERIVE functions : DERIVE1 , DERIVE2 , DERIVE3 and DERIVE4. Together they can handle any function having from one to four arguments. A sample use of DERIVE1 is as follows:

FUNCT = DERIVE1 (DATATABLE , MACHT)

where DATATABLE is the starting location of the memory table containing the data which were were placed there earlier by the DATASTORE subroutine. MACHT is an argument table containing FRAC in the first word (see figure 6-2 for the significance of FRAC), the breakpoint pointer in the next half-word and the number of Mach breakpoints in the last half-word.

MACHT	FRAC	
	POINTER	MACHT LENGTH

For our existing example, the pitching derivative bias CM0 appears at last, on the right-hand side of an assignment statement

```
CM0 = DERIVE2 ( CM0T , ALPHAT , MACHT )
```

7. REPRESENTATIVE SOLUTIONS

No simulation can be completely trusted unless it has undergone preliminary STATIC and DYNAMIC CHECKS. It is imperative that such checks be carried out before the very first production run and especially if the simulation is just started up after some lay-off time. Check conditions need not be unique; any reasonable set of conditions will suffice. However, some considerations should be given to checking conditions commonly encountered in the flight regime.

7.1 General Procedures

The STATIC CHECK for the rotational equations could be performed as follows: an initial flight condition is established by fixing Mach number, dynamic pressure and angle of attack at pre-selected values. Input variables, angular rates, Euler angles, control surfaces, etc., to each of the equations are then perturbed one at a time or in pairs to excite particular terms of the equations to be checked. The outputs should be scrutinized for trends and for possible wrong signs.

By choice of certain parameters we can isolate each portion of an equation to see if it gives the correct answer. A case in point is illustrated by considering

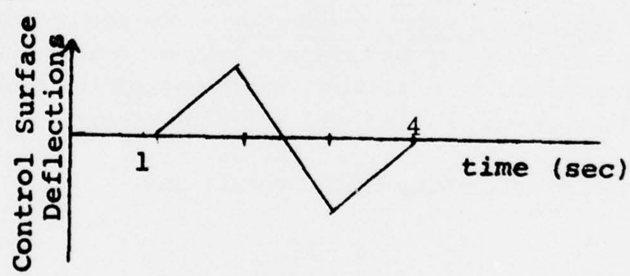
$$\dot{P}_B = \frac{\bar{q} S b}{I_X}\left[\left(C_{\ell_p} P_B + C_{\ell_R} R_B\right)\frac{b}{2V} + C_{\ell_\beta}\beta + C_{\ell_{\delta a}}\delta a + C_{\ell_{\delta r}}\delta r\right]$$

$$+ \frac{I_{XZ}}{I_X}\left(\dot{R}_B + P_B Q_B\right) + \left(\frac{I_Y - I_Z}{I_X}\right) Q_B R_B \qquad (7\text{-}1)$$

If we set $P_B = Q_B = 0$ and $\beta = \delta_a = \delta_r = I_{XZ} = 0$ we can observe the change in roll rate as a sole function of the yaw rate, or $\dot{P}_B = f(R_B)$. By resetting different parameters, we can gradually examine the influence of each of the terms in the $\dot{P}_B$ equation, remembering to take into account the sign of each term.

A similar procedure is used to check the aerodynamic coefficient equations for C_N , C_c , C_ℓ , C_m , C_n and C_Y as functions of α , β , Mach number and control surface configuration.

DYNAMIC CHECKS should be attempted only after the user is fully satisfied with the static tests. Again a few checks are suggested herein for illustration purposes. They are neither unique nor exhaustive. To test the dynamic behavior of the rotational equations, a stable flight condition is picked and initialized by setting α, β , δe_{TRIM} , δr_{TRIM} , δa_{TRIM} , V and h (TRIM conditions are selected so that the initial moments are zero, i.e., α and β remain unchanged). Each surface (δ_e , δ_r , δ_a) is pulsed separately and the resulting aerodynamic responses on P_B , $\dot{P}_B$, Q_B , $\dot{Q}_B$, R_B , $\dot{R}_B$, α , β , acceleration terms NZ & NY , θ, φ and ψ are recorded on a stripchart and compared to known correct responses. The suggested shape of the bidirectional ramp shown below could be used for forcing function (deflection varies from $\pm 5^o$ for subsonic region to $\pm 20^o$ for supersonic region)



It is recommended that this test be conducted on a daily basis to verify the operation worthiness of the equipment. DYNAMIC CHECKS should be carried out on the translational (orbital) equations also. If all aerodynamic effects are nullified by zeroing C_c , C_Y , C_N , L_B, M_B, N_B , $\dot{P}_B$, $\dot{Q}_B$, $\dot{R}_B$ at the same time fixing α , β , θ, ψ and φ, the vehicle can then be considered as a point mass. Additional checks of the orbital equations are also performed by initializing C_c and C_N such

that the aerodynamic effects of these equations can be assessed. It should be noted that dynamic checks for handling qualities are normally performed in five degrees-of-freedom. Dynamic checks for performance (point-mass) are performed in three and six degrees-of-freedom.

The control system mechanization is checked by varying the input variables (control stick, rudder pedal, angular rates, angle of attack, Mach number, dynamic pressure, etc) and observing the control surface responses.

The cockpit instruments, switches, control stick and rudder pedals must be checked for proper mechanization (magnitude and sign).

7.2 Static Checks

The values being established for Static Check can be completely arbitrary but they must be reasonable. For instance, we can't choose a test value for $\alpha = -10^\circ$ because the Shuttle will not operate in that range. On the other hand one can pick a test value of $\alpha = 28^\circ$ or $\alpha = 30^\circ$ and one value is as good as the other. In this program the Orbiter can assume either the ENTRY phase or the TAEM phase. Each phase should be tested individually.

7.2.1 Equations-of-Motion Checks: By setting certain flags the FCS can be effectively bypassed to allow us to check only the equations of motion. Specifically, to disengage the FCS, reset the flags concerning aileron and rudder and trim the elevon $\delta_{e_t} = -16.33^\circ$

Then establish the following initial conditions

Mach = 3	$\alpha = 20$	$\bar{q} = 150$
h = 99480	V = 2970	S = 2690
b = 78.06	$\bar{c} = 39.57$	$W_0 = 155{,}000$
$I_X = 600{,}000$	$I_Y = I_Z = 5.5 * 10^6$	$I_{XZ} = 10^5$
$\bar{x} = 65\%$	$\bar{z} = 29.06\%$	$\delta_{BF} = -5^\circ$
$\delta_{SB} = 25^\circ$	Gear in Up position	$\delta_e = -16.33^\circ$

With the above IC's, equation (2-19) become,

$$\dot{P}_B = \left(\frac{I_Y - I_Z}{I_X}\right) Q_B R_B + \frac{I_{XZ}}{I_X}\left(\dot{R}_B + P_B Q_B\right) + \frac{\bar{q} S b}{I_X}\Big[\left(C_{\ell_\beta} + C_{\ell_{\beta_{\delta e1}}} \delta_e\right)\beta$$

$$+ C_{\ell_{\delta a}} \delta a + C_{\ell_{\delta r}} \delta r\Big] + \frac{\bar{q} S b^2}{2 V I_X}\left(C_{\ell_P} P_B + C_{\ell_R} R_B\right) \qquad (7\text{-}2)$$

Substituting known quantities into (8-2)

$$\dot{P}_B = .1667\left(\dot{R}_B + P_B Q_B\right) + 52.495\Big[\left(-.00168 - .00001 * (-16.33)\right)\beta$$
$$+ .000547\,\delta_a + .00022\,\delta_r\Big] + .68986\left(-.261 P_B + .073 R_B\right) \qquad (7\text{-}3)$$

Next we manipulate the state variables to obtain a number of conditions. The series of checks appear lengthy at first glance, but they are necessary , because each set of conditions allows us to verify a different part of the set of equations of motion.

a/ For $P_B = Q_B = 0$; $R_B = .4$ rad/sec ; $\beta = 0^\circ$
$I_{XZ} = 0$ resulting in $\dot{P}_B = .0201$ rad/sec^2

b/ For $Q_B = R_B = 0$; $\beta = \delta_a = \delta_r = 0$; $P_B = 1.$
$I_{XZ} = 0$ $\dot{P}_B = -.1801$

c/ For $P_B = Q_B = R_B = 0$; $\beta = -5^\circ$; $\delta_a = \delta_r = 0$
$I_{XZ} = 0$ $\dot{P}_B = .3990$

d/ For $P_B = Q_B = R_B = 0$; $\delta_a = 10^\circ$; $\beta = \delta_r = 0$
$I_{XZ} = 0$ $\dot{P}_B = .2871$

e/ For $P_B = Q_B = R_B = 0$; $\delta_r = 20^\circ$; $\beta = \delta_a = 0$
$I_{XZ} = 0$ $\dot{P}_B = .2310$

f/ For $P_B = 0$; $Q_B = R_B = .4$; $\beta = \delta_a = \delta_r = 0$
$I_{XZ} = 0$ $\dot{P}_B = .1534$

In the next two cases, g) and h), I_{XZ} is not set to zero but to 10^5 as indicated earlier.

g/ For $P_B = Q_B = R_B = 0$; $\delta_a = 10^o$; $\beta = \delta_r = 0$

$\dot{R}_B = .0058$ Resulting in $\dot{P}_B = .2881$

h/ For $P_B = 1$; $Q_B = .4$; $R_B = 0$; $\delta_a = 10^o$; $\beta = \delta_r = 0$

$\dot{R}_B = -.3541$ $\dot{P}_B = .1148$

Still under the IC's imposed earlier, calculate the static test concerning the pitch acceleration. According to equation (2-19)

$$\dot{Q}_B = \left(\frac{I_Z - I_X}{I_Y}\right) P_B R_B + \frac{I_{XZ}}{I_Y}\left(R_B^2 - P_B^2\right) + \frac{\bar{q} S \bar{c}}{I_Y}\Big[C_{m_0} + \Delta C_{m_e} +$$
$$C_{m_{\delta BF1}} \delta_{BF} + C_{m_{\delta SB1}} (\delta_{SB} - 55^o)\Big] + \frac{\bar{q} S \bar{c}^2}{2 V I_Y}\left(C_{m_Q} Q_B\right) \qquad (7\text{-}4)$$

Plugging in the inertia terms

$$\dot{Q}_B = .8909\, P_B R_B + .0182\, (R_B^2 - P_B^2) + 2.903\Big[-.023 + \Delta C_{m_e} - .00051\, \delta_{BF}$$
$$+ .00028\, (\delta_{SB} - 55^o)\Big] + .0193\, (-2.3\, Q_B) \qquad (7\text{-}5)$$

As a consequence

i/ For $P_B = Q_B = R_B = 0$; $\delta_e = 0$; $\delta_{BF} = -5^o$; $\delta_{SB} = 25^o$

Whereupon the check value for the pitch acceleration is

$\dot{Q}_B = -.0838$ rad/sec^2

j/ For $P_B = Q_B = R_B = 0$; $\delta_e = -16.33^o$; $\delta_{BF} = 16.3^o$

$\delta_{SB} = 25^o$ $\dot{Q}_B = -.0831$

k/ For $P_B = Q_B = R_B = 0$; $\delta_e = -16.33^o$; $\delta_{BF} = -5^o$

$\delta_{SB} = 87.2^o$ $\dot{Q}_B = .072$

l/ For $P_B = R_B = 0$; $Q_B = .5$; $\delta_e = -16.33^o$; $\delta_{BF} = -5^o$

$\delta_{SB} = 25^o$ $\qquad \dot{Q}_B = -.0222$

m/ For $P_B = Q_B = 0$; $R_B = .4$; $\delta_e = -16.33^o$; $\delta_{BF} = -5^o$

$\delta_{SB} = 25^o$ $\qquad \dot{Q}_B = .0029$

n/ For $P_B = 1.$; $Q_B = 0$; $R_B = .4$; $\delta_e = -16.33^o$

$\delta_{BF} = -5^o$; $\delta_{SB} = 25^o$ $\qquad \dot{Q}_B = .3411$

Next let us turn our attention to the yaw acceleration by making use of equation (2-19) again

$$\dot{R}_B = \left(\frac{I_X - I_Y}{I_Z}\right) P_B Q_B + \frac{I_{XZ}}{I_Z}\left(\dot{P}_B - Q_B R_B\right) + \frac{\bar{q}\,S\,b}{I_Z}\Big[(C_{n_\beta} + C_{n_{\beta\delta e}}\,\delta e)\beta + C_{n_{\delta a}}\,\delta a$$

$$+ \left(C_{n_{\delta r}}\,\delta r + C_{n_{\delta r \beta}}|\beta|)\delta r\right] + \frac{\bar{q}\,S\,b^2}{2\,V\,I_Z}\left(C_{nP}\,P_B + C_{nR}\,R_B\right) \qquad (7\text{-}6)$$

$$\dot{R}_B = -.8909\,P_B Q_B + .0182\,(\dot{P}_B - Q_B R_B) + 5.727\,\Big[-.00096\,\beta + .00001\,\delta_a$$

$$+ (-.00042 + .000016\,|\beta|)\,\delta_r\Big] + .0753\,\left[.01\,P_B - .91\,R_B\right]$$

Carrying on with the test

o/ For $P_B = Q_B = 0$; $R_B = .4$; $\beta = \delta_a = \delta_r = 0$

$I_{XZ} = 0$, we obtain $\qquad \dot{R}_B = -.0274$

p/ For $P_B = 1.$; $Q_B = R_B = 0$; $\beta = \delta_a = \delta_r = 0$

$I_{XZ} = 0$ $\qquad \dot{R}_B = .0008$

q/ For $P_B = Q_B = R_B = 0$; $\beta = -5^o$; $\delta_a = \delta_r = 0$

$I_{XZ} = 0$ $\qquad \dot{R}_B = .0275$

r/ For $P_B = Q_B = R_B = 0$; $\delta_a = 20^o$; $\delta_r = \beta = 0$

$I_{XZ} = 0$ $\dot{R}_B = .0011$

s/ For $P_B = Q_B = R_B = 0$; $\delta_r = 20^o$; $\beta = \delta_a = 0$

$I_{XZ} = 0$ $\dot{R}_B = -.0481$

t/ For $P_B = 1.$; $Q_B = .4$; $R_B = 0$; $\beta = \delta_a = \delta_r = 0$

$I_{XZ} = 0$ $\dot{R}_B = -.3556$

Now we again impose a value of 10^5 on I_{XZ}

u/ For $P_B = Q_B = R_B = 0$; $\delta_a = 20^o$; $\beta = \delta_r = 0$

$\dot{P}_B = .5761$ rad/sec^2 $\dot{R}_B = .0116$

v/ For $P_B = 0$; $Q_B = R_B = .4$; $\delta_a = 20$; $\beta = \delta_r = 0$

$\dot{P}_B = .5913$ $\dot{R}_B = -.0184$

7.2.2 FCS Checks : Set ALTITUDE = 92,800 ft ; VELTRUE = 2960 ft/sec ; TURNCORD = .TRUE. (to allow the FCS to maintain the nose level during a turn) and Rudder Pedal Position RPTASOP = 0. Then proceed to parts a) and b) below.

a) Pitch Plane Test (ENTRY Phase) : Fix the Pitch RHC , PRHCSOP = -10^o and $Q_B = 2^o$/sec. Then the elevon deflection, the final object of our Pitch-plane test, should be $\delta_e = -6.42^o$. The calculation of intermediate variables leading to δ_e is left to the reader.

b) Roll and Yaw Test (ENTRY Phase) : Establish the Roll Stick Position RRHCSOP = 16.15^o and also the following conditions

PRHCSOP = 0^o	$P_B = 5^o$/sec
$\varphi = 30^o$	$R_B = 1^o$/sec
$\theta = 20^o$	$\alpha = 15^o$

We should obtain these final results

$$\delta_a = -2.66^\circ$$

$$\delta_r = -1.9^\circ$$

and Yaw-jet Command DRJET = .63 (dimensionless) corresponding to the firing of two reaction jets (UZCMD = 2).

Now we are ready to test the FCS in TAEM phase. Assume ALTITUDE = 48,600 ft and VELTRUE = 1017 ft/sec. This will result in Mach = 1.05 and $\bar{q} = 200\ lb/ft^2$. Proceed to parts c) and d) next.

c) Pitch-Plane Test (TAEM Phase) : Choose PRHCSOP = -10° ; RRHCSOP = 0° ; RPTASOP = 0° ; $Q_B = -2^\circ/sec$; $R_B = -2^\circ/sec$ and $\varphi = 30^\circ$. With the above initial conditions we should command the following responses:

Pitch-jet command, DPJET = -.79
Pitch command signal, BCSL = .365

d) Roll & Yaw Test (TAEM Phase) : Select the conditions listed below.

PRHCSOP = 0°	RRHCSOP = 11.15°
RPTASOP = 11.125°	$P_B = 2^\circ/sec$
$\varphi = 30^\circ$	$R_B = 5^\circ/sec$
$\theta = 20^\circ$	NY = .25 g
$\alpha = 15^\circ$	$\dot{R}_B = 0$

Look for the following results out of the FCS

$$\delta_a = .771^\circ$$

$$\delta_r = 22.09^\circ$$

DRJET = -10.68 (4 jets)

7.3 Dynamic Checks

Again, two principal parts of the program are tested semi-independently of each other. We choose to test the simulation in 5 DOF's only, by setting FIVDEG = .TRUE. . In 5-DOF mode the Shuttle flies at a constant altitude at a constant axial velocity. To disturb the equilibrium (or steady-state) conditions, we introduce a forcing-function of short duration and observe the system response. The triangular-waveform forcing-function shown in paragraph 7.1 is generated on the analog computer and patched into an A-D convertor. It has its peaks at ± 1 . Once converted into a digital quantity, this waveform is further modulated to give the proper pulse amplitude (for example $\pm 16^o$ for δ_e pulse). During each test only one surface is disturbed. All other surface deflections are returned to initial-condition values or set to zero. Both the equations-of-motion test and the FCS test use the same standard conditions for altitude, true airspeed, weight, surface area, inertial coefficients, Euler angles and trim surface deflections, namely

$I_X = 6*10^5$	$\bar{x} = .65$	$V_T = 2970$
$I_Y = 5.5*10^6$	$\bar{y} = 0$	$\bar{q} = 150$
$I_Z = 5.5*10^6$	$\bar{z} = .2907$	Mach = 3
$I_{XY} = 0$	h = 99,480	S = 2690
$I_{XZ} = 10^5$	$W_0 = 1.55*10^5$	b = 78.03
$\bar{c} = 39.56$	$\ell_r = 107.5$	

The following angular conditions are also arbitrarily imposed

$\delta_{SB} = 25^o$	$\varphi(0) = 0^o$	$L(0) = 0^o$
$\delta_{BF} = -5^o$	$\Psi(0) = 0^o$	$\lambda(0) = 0^o$
$\alpha(0) = 20^o$	$\theta(0) = 20^o$	$\delta_a = 0^o$
$\beta(0) = 0^o$	$\delta_e = -16.33^o$	$\delta_r = 0^o$

7.3.1 Equations-of-Motion Tests : Three runs are made , all with the following conditions:

FCSDE = FCSDA = FCSDR = .TRUE.

Various output quantities are fed to the DAC's and appear on the strip charts as analog signals. Under the best circumstances the traces can be read at an accuracy of 1% and thus can be used as a qualitative check only. Distinguish the three runs:

- Case a : δ_e is pulsed with a triangular wave (see page 65) having $\pm 16^o$ peaks . See figure 7-1.

- Case b: δ_a is pulsed with the same waveform resulting in the traces on figure 7-2.

- Case c: δ_r is pulsed with a triangular signal having $\pm 20^o$ peaks. Consult figure 7-3 for the airframe responses.

7.3.2 Flight-Control-System Tests: The FCS tests helps to verify the end-to-end performance of the simulation. Since we operate at Mach 3 and since rudder pedal inputs are bypassed by the FCS at Mach > 1.5, there is no need to pulse the rudder surface. In the two remaining test cases, a full stick input is applied to the aileron in one case and to the elevon in the other case. The testing is intended not only for dynamic system response, but also in assisting in the discovery of possible FCS discontinuities or errors. Both cases are made with the following conditions:

FCSDE = FCSDA = FCSDR = .FALSE.

As before, the system response is observed through the use of strip-chart recorders. The two cases are perturbated in the following manner:

- Case d : $\delta_{e_{RHC}}$ is pulsed with a triangular wave having peaks at $\pm 20^o$. See figure 7-4.

- Case e: $\delta_{a_{RHC}}$ is pulsed by the same signal. See figure 7-5.

It should be noted that in both cases the Flight Control System must be allowed to stabilize the airframe for approximately 3 to 5 seconds before applying the disturbances. In this manner the pitch axis will settle to a trim condition, reducing the inter-axis interaction.

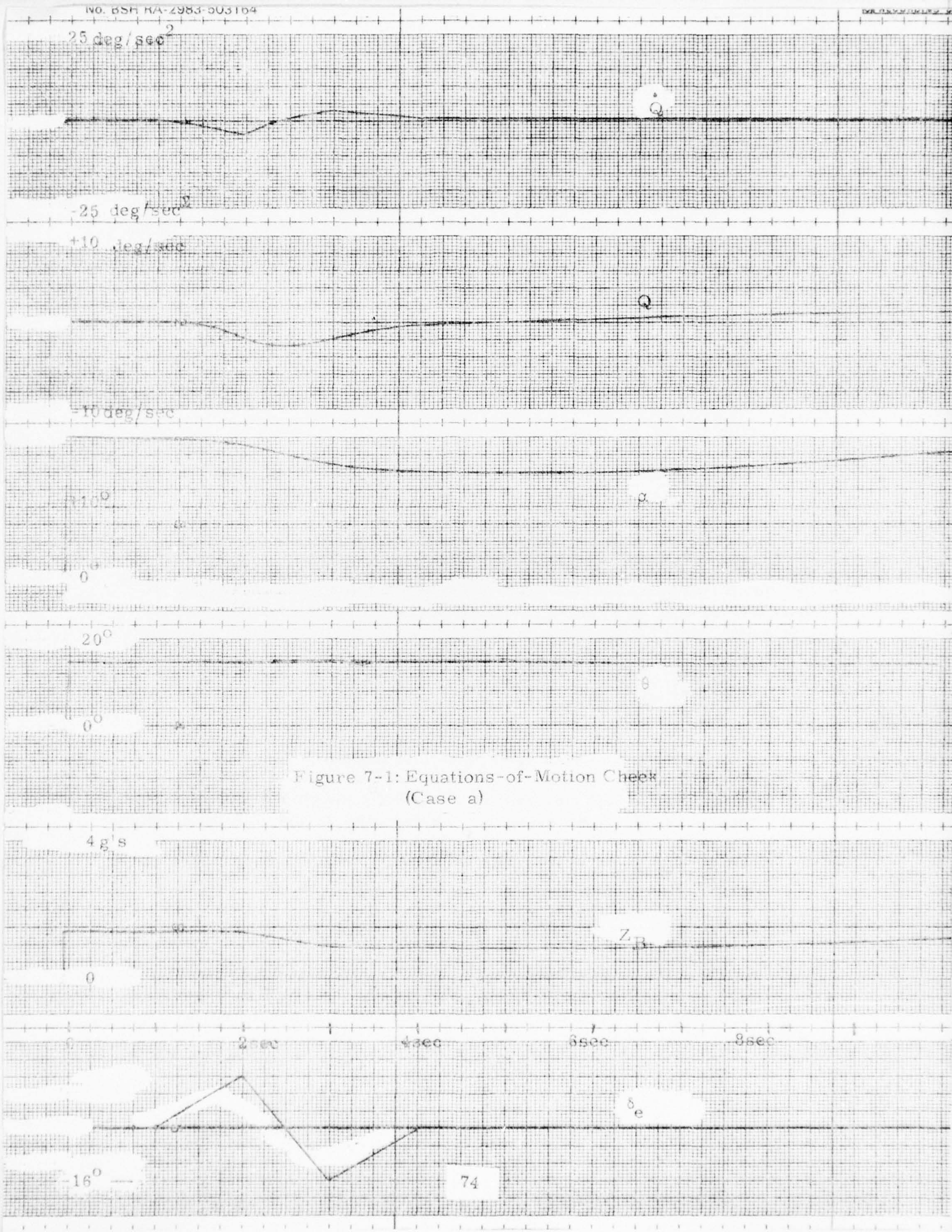
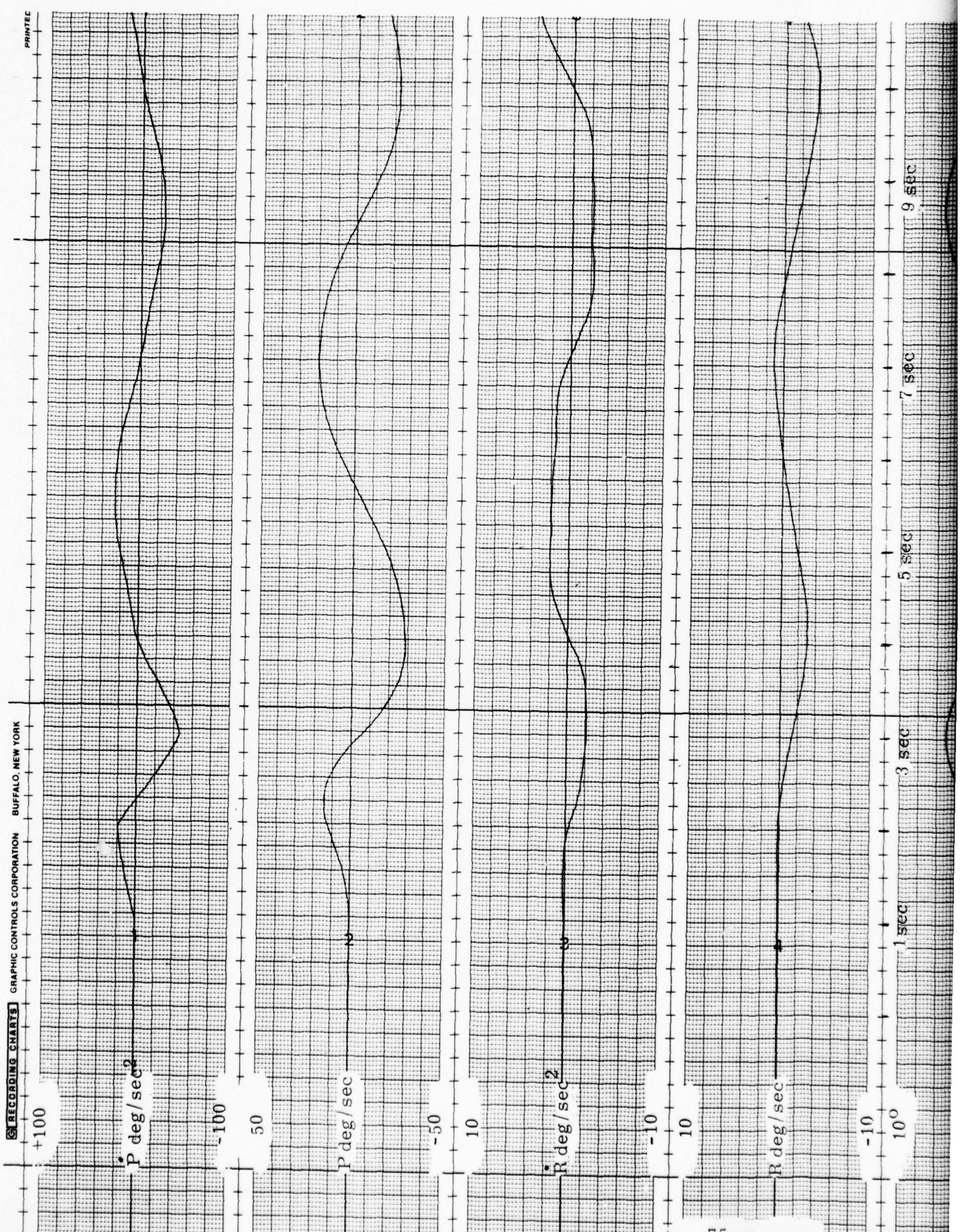


Figure 7-1: Equations-of-Motion Check
(Case a)



RECORDING CHARTS GRAPHIC CONTROLS CORPORATION BUFFALO, NEW YORK
PRINTED
+100
P deg/sec2
-100
50
P deg/sec
-50
10
R deg/sec2
-10
10
R deg/sec
-10
10°
1
2
3
4
1 sec
3 sec
5 sec
7 sec
9 sec

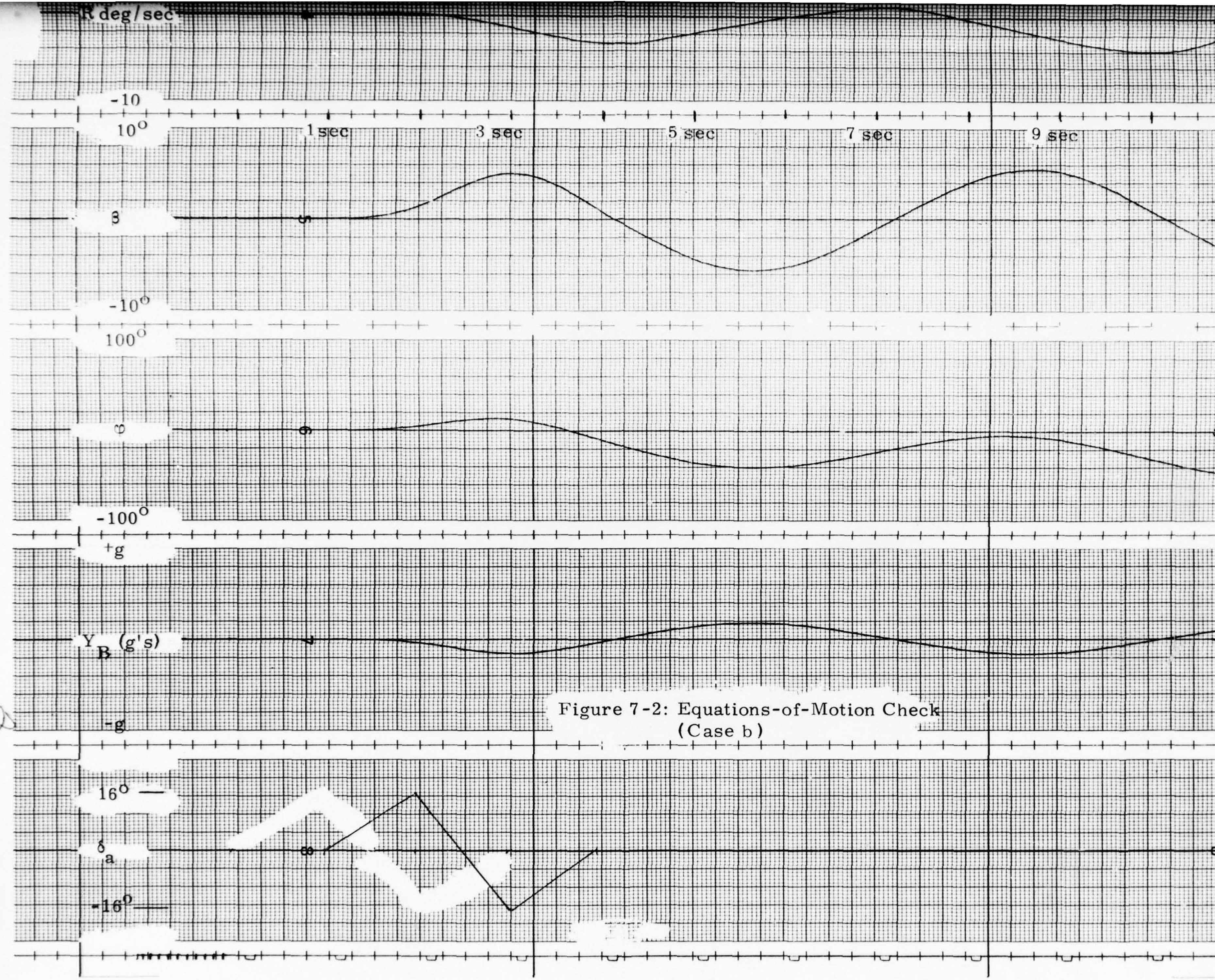


Figure 7-2: Equations-of-Motion Check
(Case b)

2

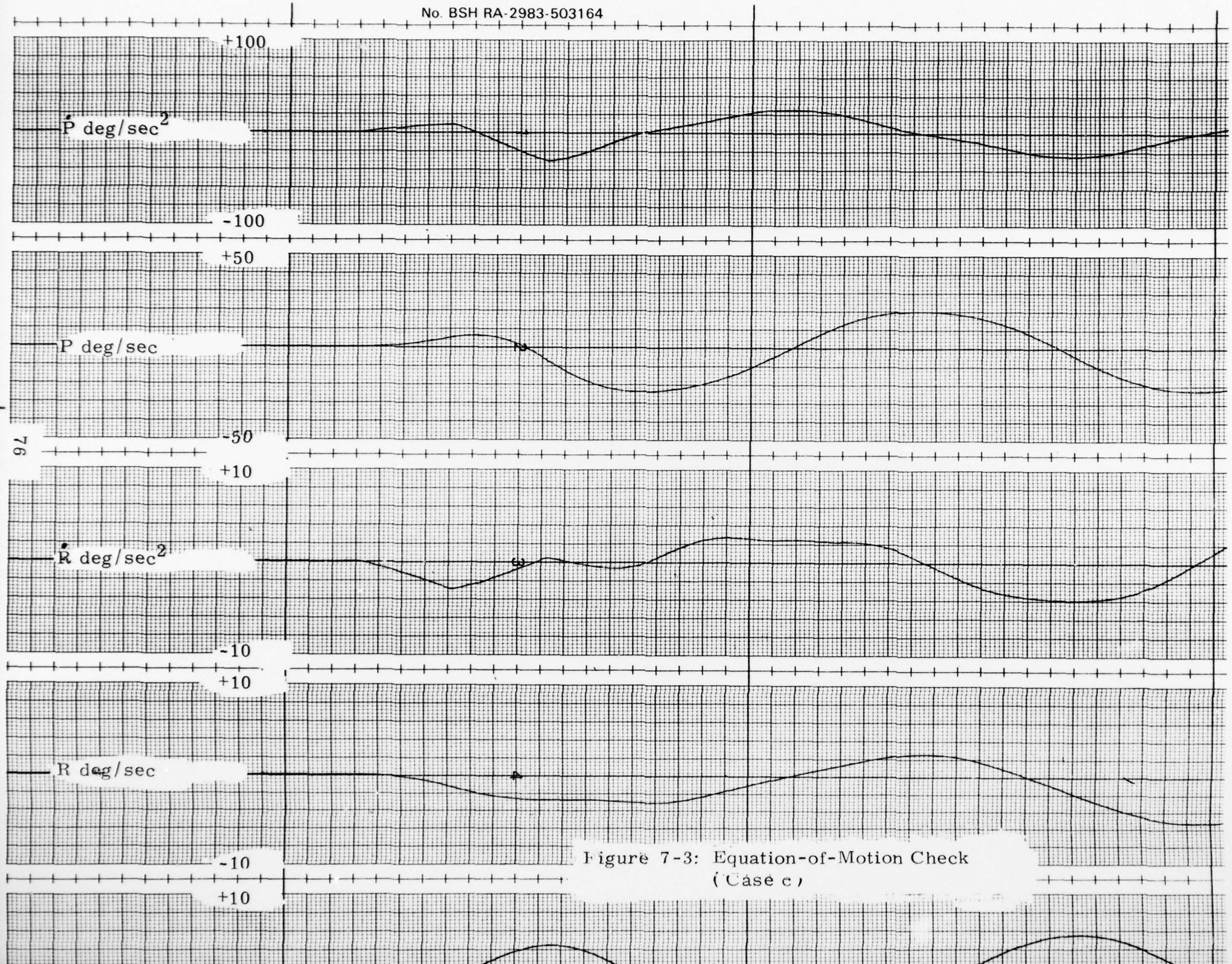


Figure 7-3: Equation-of-Motion Check
(Case c)

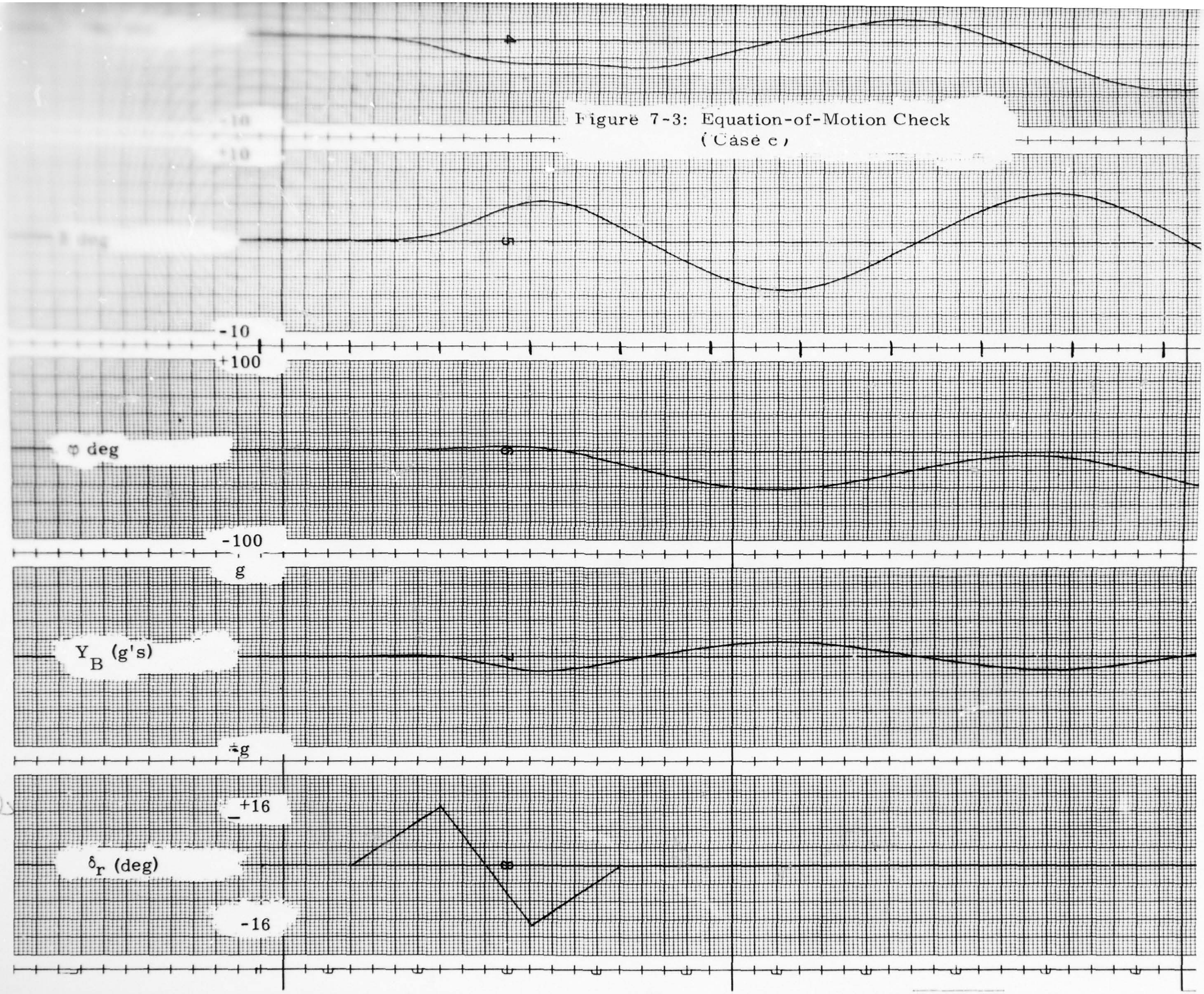


Figure 7-3: Equation-of-Motion Check
(Case c)

2

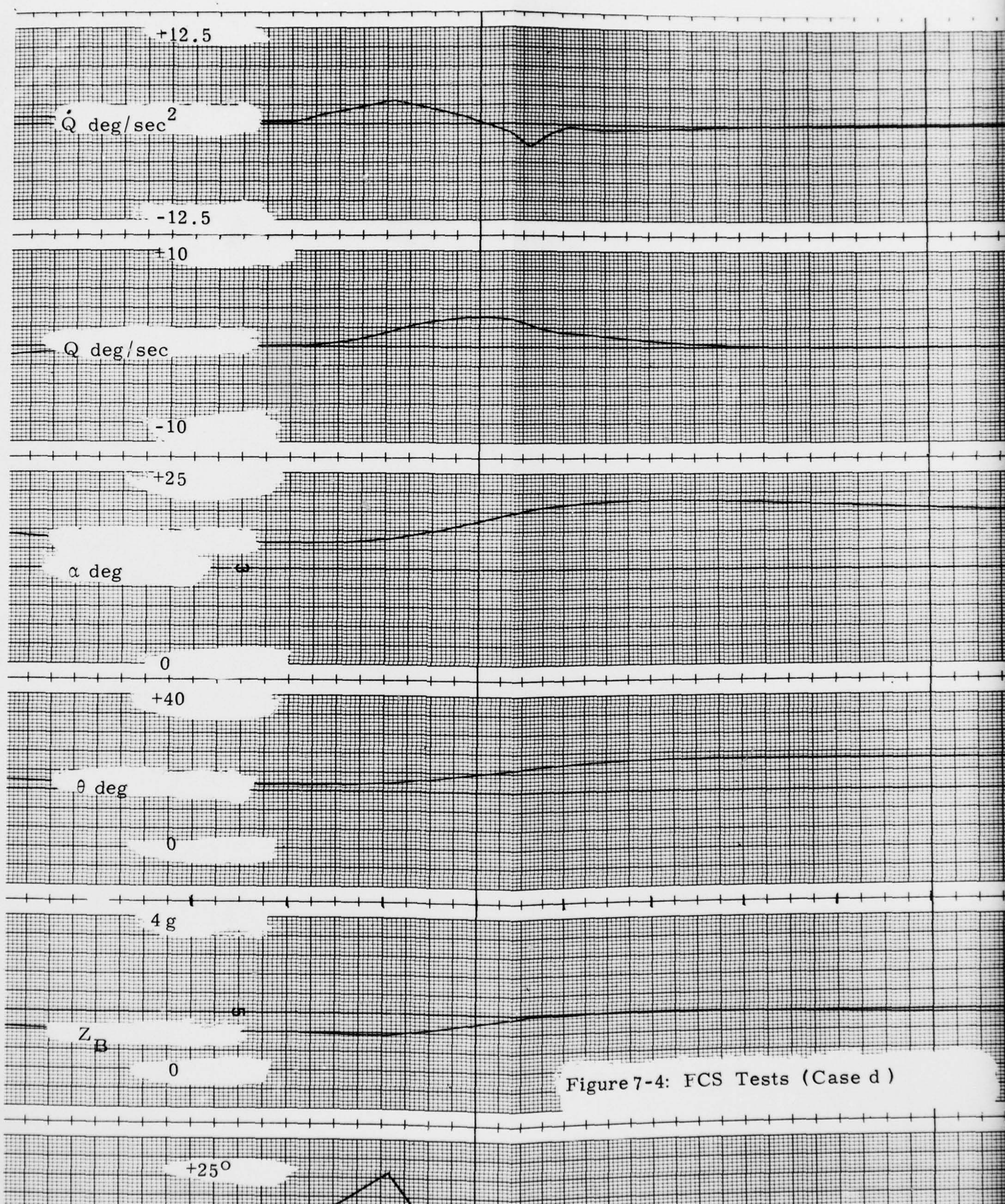
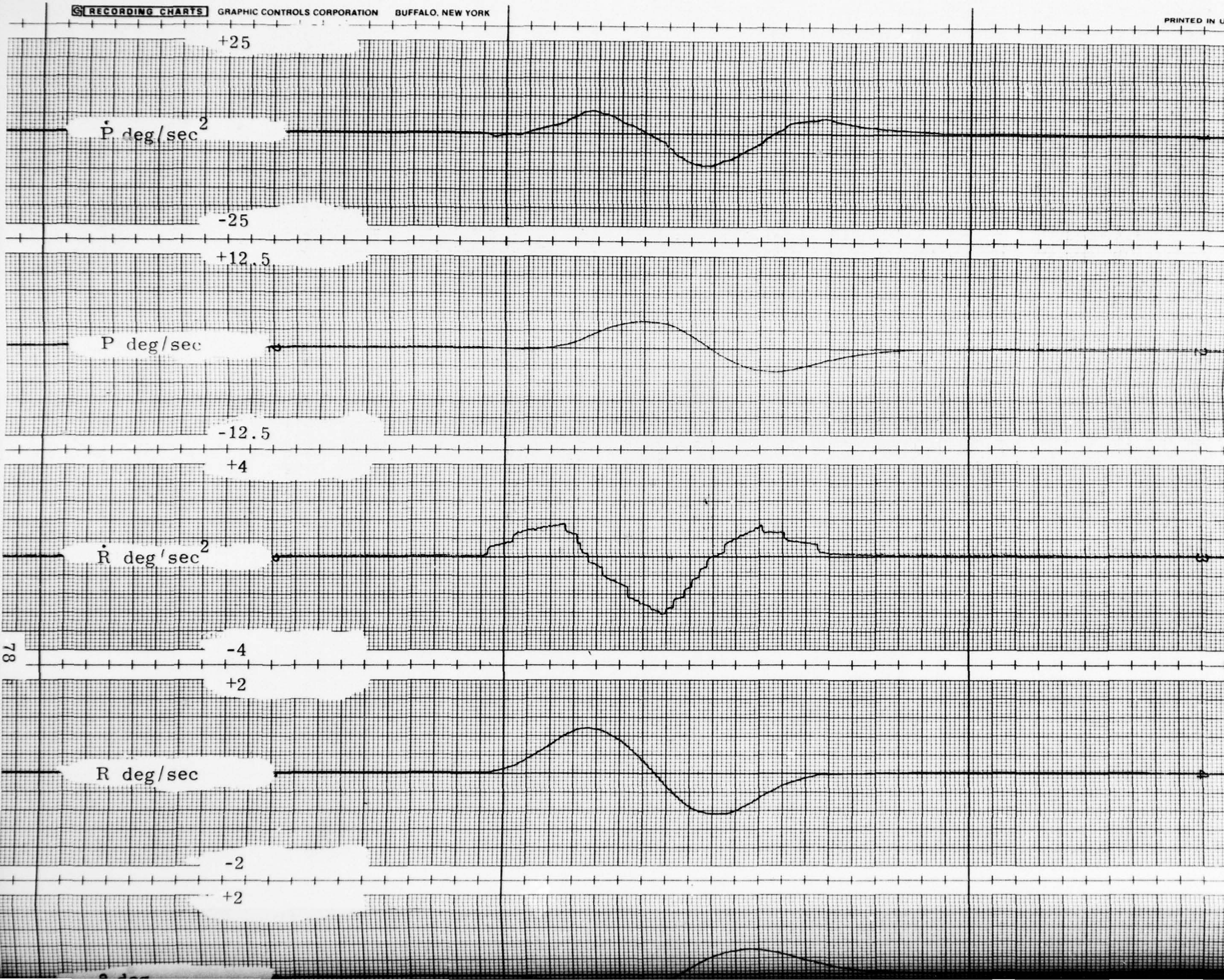


Figure 7-4: FCS Tests (Case d)


RECORDING CHARTS GRAPHIC CONTROLS CORPORATION BUFFALO, NEW YORK
PRINTED IN U.S
+25
$\dot{P}$ deg/sec^2
-25
+12.5
P deg/sec
-12.5
+4
$\dot{R}$ deg/sec^2
-4
+2
R deg/sec
-2
+2

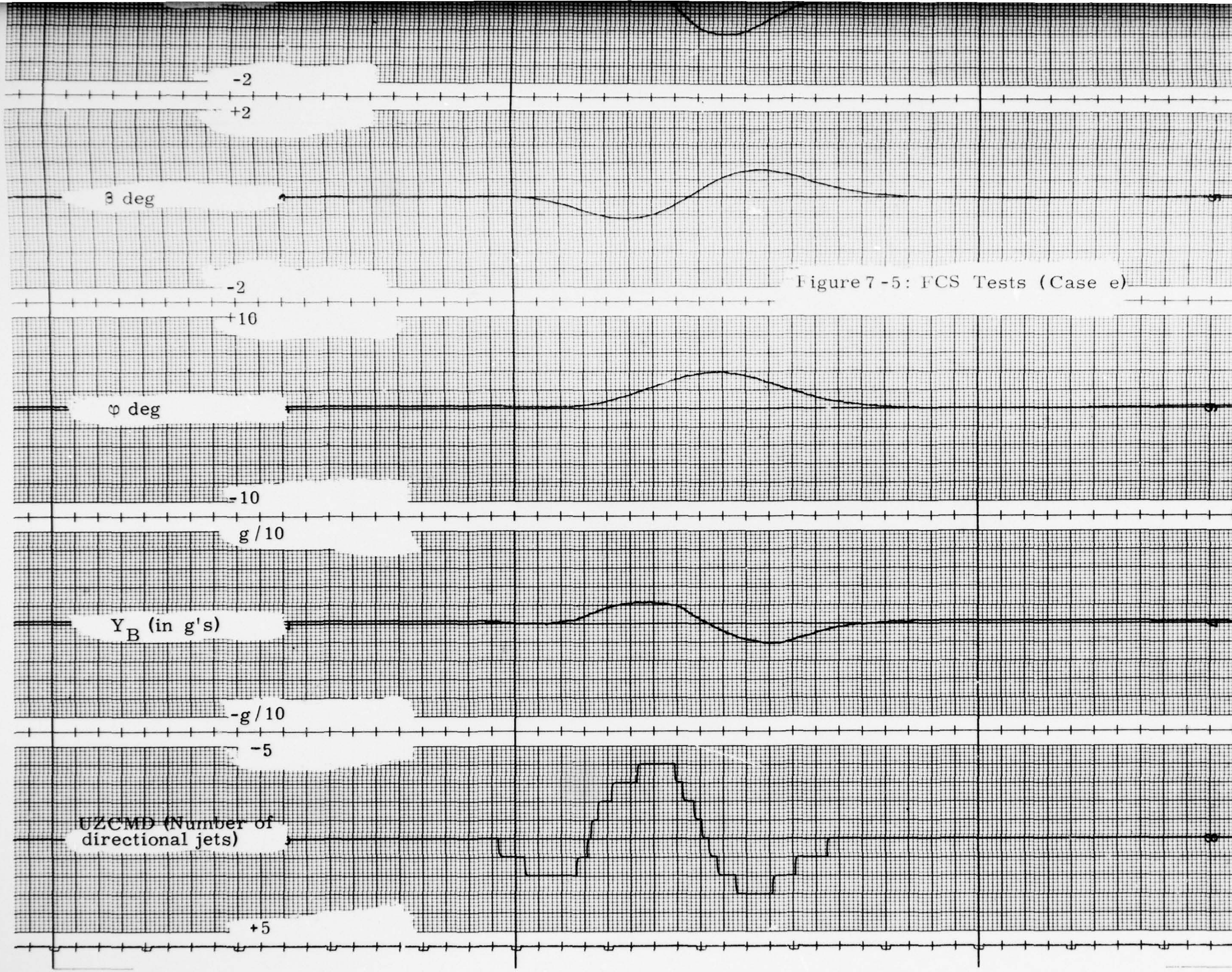


Figure 7-5: FCS Tests (Case e)

2

8. TERMINOLOGY, SYMBOLS AND DEFINITIONS

8.1 List of Acronyms

ASCII	American Standard Code for Information Interchange
A-D	Analog-to-Digital
CG	Center of Gravity
CMD	Command ; Commanded
CRT	Cathode-Ray Tube (Computer terminal)
C/R	Carriage Return
DAC	Digital-to-Analog Convertor
DOF	Degree Of Freedom
EOF	End-Of-File
FCS	Flight Control System
FGP	Function-Generation Package
FILT	Filter
Hex	Hexadecimal number
IC	Initial Condition mode
I/O	Input / Output
NOM	Nominal value
RHC	Rotational Hand Control
RCS	Reaction-jet Control System
SEL	Systems Engineering Laboratories

8.2 Lists of Symbols Used in Computer Programs

The following abbreviation is used throughout this chapter: DL = dimensionless quantity. The subscripts on various quantities have the following significance:

δa = aileron incremental deflection
B = written in Body coordinate system
δBF = Body Flap incremental deflection

δe = elevon incremental deflection
δr = rudder " "
δSB = Speed Brake incremental deflection

E or e = Earth axes or relating to earth
F = Fuel
G = Gear extended

H = written in H-frame (orbital reference axes)
L, R = two halves (Left & Right) of the elevon
0 = bias value ; IC value ; sea-level value

Be aware that stability coefficients with various final subscripts may have different units. Consider C_{N_0}, $C_{N_{\delta e}}$, $C_{N_{\delta BF}}$ (bias value of normal force coefficient, normal force derivatives due to elevon and body flap incremental deflection) ; they will not appear as three separate entries on the table. Note that C_{N_0} is dimensionless while $C_{N_{\delta e}}$ and $N_{\delta BF}$ have units of sec^{-1}. Other subscripts used are α, β, p, q and r.

Sometimes a quantity is represented by two computer variables which are not equivalent. For instance ALPHA stands for α in radians. At the same time ALFA also symbolizes α but in degrees. Similarly LAT and LA both stand for latitude, but LAT is a single-precision quantity while LA is computed in double precision. The computer symbols appear on two separate tables. The second table lists the names used in the FCS program and the first table covers the remaining programs (MAIN plus all subroutines)

Table 8-1 : MAIN PROGRAM SYMBOLS

Symbol	Explanation	Computer Representation	Unit
a	Sonic speed	SOUND	ft/sec
b	Wing span	B	ft
C_c	Total trim chord coefficient	CC	DL
cg_X	Longitudinal distance of CG measured from a reference point situated 238" in front of the nose (expressed in % of Orbiter's lenght)	(not used)	%
C_ℓ	Total rolling-moment derivative	CL	DL
$\bar{c}$	Mean aerodynamic chord	C	ft
$C_{m_{\delta e}}$	Pitching-moment derivative due to δ_e	CMDE	deg^{-1}
$C_{m_{\delta SB}}$	Pitching-moment derivative due to δ_{SB}	CMSB	deg^{-1}
$C_{n_{\delta r}}$	Yawing-moment coeff. due to rudder	CNDR	deg^{-1}
C_N	Total normal force derivative	CN	DL
C_{Y_β}	Side-force coefficient due to β	CYB	deg^{-1}
F_T	Fuel consumption rate	(not used)	1/sec
g	Gravitational acceleration	GRAVITY or G	ft/sec^2
h	Geometric altitude	ALTITUDE	ft
h_p	Geopotential altitude (used for calculating air density & temperature)	GEOALT	ft
I_{XX}	Polar moment of inertia (along the longitudinal axis). Or I_X	IXX	$slug.ft^2$
I_{XY}, I_{XZ}, I_{YZ}	Products of inertia	IXY, IXZ, IYZ	$slug.ft^2$

Table 8-1 : Continued

Symbol	Explanation	Computer Representation	Unit
$I_{YY}; I_{ZZ}$	Transverse moments of inertia	IYY ; IZZ	$slug.ft^2$
KCAS	Calibrated airspeed	KCAS	knot
KEAS	Equivalent airspeed	KEAS	knot
J_2	A constant characteristic of the earth mass distribution	J2	DL
J_P, J_R, J_Y	Constants characteristic of the thrusts of the pitch, roll and yaw reaction jets.	UYCMD, UXCMD, UZCMD	DL
L	Latitude	LAT OR LA	radian
L_B	Total rolling-moment in body axes	LBODY	lb. ft
$\ell_1 ; \ell_2 ; \ell_3$	Direction cosines	L1 ; L2 ; L3	DL
ℓ_r	Length of the Orbiter (nose to tail)	LENGTH	ft
m	Orbiter's mass	MASS	slug
$m_1 ; m_2 ; m_3$	Direction cosines	M1 ; M2 ; M3	DL
M_B	Total pitching moment	MBODY	lb. ft
Mach	Mach number	MACH	DL
$n_1 ; n_2 ; n_3$	Direction cosines	N1 ; N2 ; N3	DL
N_B	Total rolling moment	NBODY	lb. ft
P_B	Rolling rate	PBODY	rad/sec
$\bar{q}$	Dynamic pressure	QBAR	lb/ft^2
Q_B	Pitching rate	QBODY	rad/sec

Table 8 -1 : Continued

Symbol	Explanation	Computer Representation	Unit
r_0	Equatorial radius of earth	R0	ft
r	Distance between Ortiter and earth center	RADIUS or RAD	ft
δR	Altitude measured from sea level	DELR	ft
R_B	Yawing rate	RBODY	rad/sec
R_D	Down range	DOWNRNG	ft
R_X	Cross range	CROSSRNG	ft
S	Reference wing surface area	S	ft^2
T	Thrust	(not used)	lb
T_X, T_Y, T_Z	Thrust vector in the 3 body axes	(not used)	lb
U_{EA}, V_{EA}	Horizontal components of airspeed in earth axes	UAIR(or UA) VAIR(or VA)	ft/sec
U_E, V_E, W_E	Orthogonal velocity components expressed in earth axes	UEARTH, VEARTH, WEARTH	ft/sec
V	Orbiter's total velocity	VELTRUE or TAS	ft/sec
W_0	Orbiter's weight	WEIGHT	lb
W_F	Fuel weight	(not used)	lb
W_u	Wind component in North direction	NWIND	ft/sec
Wv	Wind component in East direction	EWIND	ft/sec

Table 8-1 : Continued

Symbol	Explanation	Computer Representation	Unit
$\bar{x}$	Longitudinal location of Orbiter's CG (expressed in percent of Orbiter's length)	XBAR	%
X_B	Longitudinal acceleration component	XBODY, XB or AX	ft/sec^2
$\bar{y}$	Lateral location of CG	YBAR	%
Y_B	Lateral acceleration component	YBODY, YB or AY	ft/sec^2
$\bar{z}$	Vertical location of the CG	ZBAR	%
Z_B	Normal acceleration component	ZBODY, ZB or AZ	ft/sec^2
α	Angle of attack	ALPHA	radian
β	Angle of side slip	BETA	deg.
γ	Flight path angle	(not used)	rad
δ_a	Aileron total deflection	DAIL or DA	deg.
δ_{BF}	Body-flap total deflection	FLAP or DBF	deg.
δ_e	Elevon total deflection	ELEV or DE	deg.
δ_r	Rudder total deflection	DRUD or DR	deg.
δ_{SB}	Speed brake total deflection	BRAK or DSB	deg.
Δt	Time increment (time frame)	DELTA	sec
θ	Pitch angle	THETA	rad.
θ_F	Angle between surface of the fuel and X_B axis	(not used)	deg.

Table 8-1 : Continued

Symbol	Explanation	Computer Representaion	Unit
λ	Longitude	LNG or LN	rad
$\bar{\mu}$	Colatitude ($\frac{\pi}{2}$ -L)	MU	rad
ρ	Atmospheric density	RHO	slug/ft^3
φ	Roll angle	PHI	rad
Ψ	Yaw angle	PSI	rad
ω_e	Angular velocity of earth	OMEGA	rad/sec

Some variables appear both in the Main program and the SHTFCS. If they have been previously listed on Table 8-1, they will not be repeated here on Table 8-2.

Table 8-2 : FCS Variables

Computer Variable	Description	Unit
DACMD	Aileron command	deg
DATMPAN	Aileron panel trim command	0,$\pm$1
DATMRHC	Aileron stick trim command	0,$\pm$1
DATSUMI	Aileron trim integrator command	deg
DBFMAN	Manual Body Flap command	0,$\pm$1
DBFRC	Body Flap rate command	deg/sec

Table 8-2 : Continued

Computer Variable	Description	Unit
DCSP	Aileron command due to stick and rate feedback	deg
DECMD	Elevator command	deg
DEL	Left elevon deflection	deg
DER	Right elevon deflection	deg
DETRIM	Elevator position trim command	deg
DETMPAN	Elevator panel trim command	$0, \pm 1$
DETMRHC	Elevator stick trim command	$0, \pm 1$
DPJET	Pitch Jet Command	DL
DQCT	Elevator Command due to stick and rate feedback	deg
DRCPF	Rudder command due to pedal	deg
DRCMD	Rudder command	deg
DRJET	Yaw Jet Command	DL
DRLP	Left-half rudder deflection	deg
DRRP	Right-half rudder deflection	deg
DRTRIM	Yaw Trim Integrator output	deg
DSBPC	Speedbrake command	deg
ELFBK	Elevator Feedback signal (Mach$\leq$12)	deg
ETRIM	Elevator Position Trim (Mach>12)	deg
NY	Lateral Acceleration	g's
NZ	Normal Acceleration	g's

Table 8-2 : Continued

Computer Variable	Description	Unit
PE	Roll & Yaw stability feedback	DL
PRHCSOP	Pitch Stick Command	deg
QDOT	Pitch Acceleration	rad/sec^2
RDOT	Yaw Acceleration	rad/sec^2
RPTASOP	Rudder Pedal Command	deg
RRHCSOP	Roll Stick Command	deg
SBHP	Speedbrake Handle Command	deg
SUM23	Roll & Yaw Command due to stick	DL

9. REFERENCES

9.1 General Bibliography

9.1.1 Dommasch D.O., S.S., Sherby and T.F. Connolly, "Airplane Aerodynamics," Pitman Publishing Corp., 1967.

9.1.2 Etkins I., D. McRuez and D. Graham, "Aircraft Dynamics and Automatic Control," Princeton University Press, 1973.

9.1.3 Howe R.M. and L.E. Fogarty, "Flight Simulation of Orbital and Re-Entry Vehicles - Part II, A Modified Flight-Path Axis System for Solving the 6DOF Flight Equations," University of Michigan, 1961. Contract No AF33(616)-5664.

9.1.4 Nagy C.J. and D.F. Richardson, "Interpretation of Space Shuttle Orbiter Aerodynamic Data For the MAP Simulation", AFFTC (Office of Advanced Manned Vehicles) Office Memo, Edwards AFB, September 1976.

9.1.5 Nagy C.J., "A New Method for Test and Analysis of Dynamic Stability and Control, " AFFTC-TD-75-4, Edwards AFB, May 1976

9.1.6 Rabiner, L.R. and B. Gold, "Theory and Application of Digital Signal Processing", Prentice-Hall, Englewood Cliffs, New Jersey 1975.

9.2 Shuttle-Related References

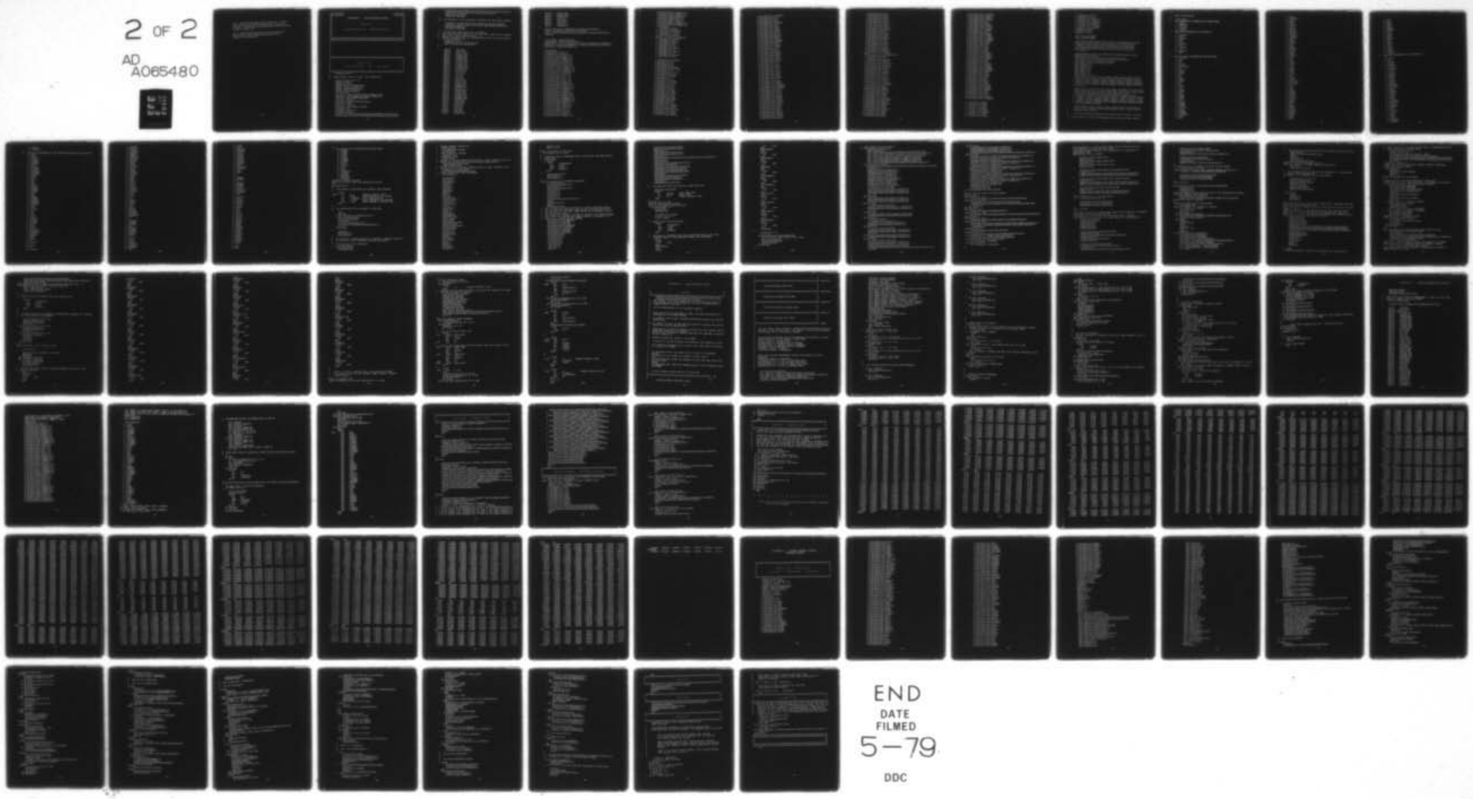
9.2.1 "Space Shuttle Flight Control System Data Book" Vol. II Orbiter, SD73-SH-0097-2D, Rockwell International Space Division Downey CA, May 1975.

9.2.2 "AFFTC Evaluation of the Space Shuttle Orbiter and Carrier Aircraft -- NASA Approach and Landing Test", AFFTC-TR-78-14, Edwards AFB, May 1978.

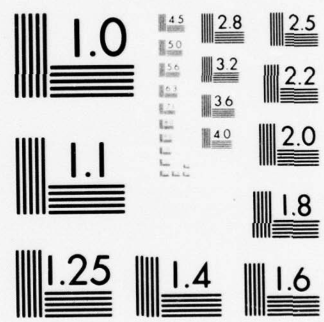
9.2.3 "Space Shuttle Orbital Flight Test -- Level C, Functional Subsystem Software Requirements, Guidance Navigation and Control, Part C, Flight Control Entry", SD76-SH-0007, Rockwell International Space Division, Downey CA, November 1976.

AD-A065 480 UNG (MAN T) EL SEGUNDO CALIF F/G 9/2
SPACE SHUTTLE DIGITAL COMPUTER SIMULATION BENCHMARK.(U)
DEC 78 M T UNG F04700-78-M-2539
UNCLASSIFIED AFFTC-TR-78-30 NL

2 OF 2
AD
A065480



END
DATE
FILMED
5-79
DDC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

9.2.4 "Space Shuttle Orbiter Orbital Flight Test -- Level C, Functional Subsystem Software Requirements, Displays and Controls", SD76-SH-0020, Rockwell International Space Division, Downey CA, March 1977.

9.2.5 "Space Shuttle Engineering Pre-Simulation Report", SD74-SH-0301, Rockwell International Space Division, Downey CA, November 1974.

APPENDIX A - MAIN PROGRAM LISTING

```
S A F T D

S I M U L A T I O N    B E N C H M A R K

S H U T T L E

E Q U A T I O N S    O F    M O T I O N

      PROGRAM SHTLOFT
*
*   SPACE SHUTTLE ORBITAL FLIGHT TEST SIMULATION.
*
       IMPLICIT REAL *4(A - Z)
       REAL*8 DATADBL
       LOGICAL*1 DATABYT
       COMMON /ARRAY1/DATAWRDR(400)
       COMMON /ARRAY2/DATADBL(17)
       COMMON /ARRAY3/DATABYT(17)
       COMMON /ARRAY4/ANGLE(3)
       BIT INBIT(16)
       LOGICAL*1 IC,HOLD,POTSET,ABORT,TREDEG,FIVDEG
       LOGICAL*1 FCSDA,FCSDE,FCSDR,FCSDF,FCSSB,WIND
       LOGICAL*1 READY,DUMPFLAG,AUTOBF
       LOGICAL*1 TURNCORD
       LOGICAL*1 DEPULSE,DAPULSE,DRPULSE
       LOGICAL*4 EASER
       LOGICAL*4 DISCRTIN
       INTEGER*2 TEMP1,TEMP2,PREVTEMP
       INTEGER*4 SAVFL
       INTEGER*4 LINK,II,JJ
       INTEGER*4 HANDLER
       REAL*8 DPIT1,DPIT2,DPIT3,COSLAT,COSMU,COSPSIH,LA,LN,MU,RAD,
     * RCOSLAT,SHARE,SINLAT,SINMU,SINPSIH,UA,UB,UE,VA,VB,VE,WB,WE,
```

```
     * WHD,XB,YB,ZB,PSIH,DELR,DELRDOT,GRAVITY,LATDOT,LNGDOT,PSIHDOT,
     * RUH,RUHDOT,UH,WH,XEARTH,XH,YEARTH,YH,ZEARTH,ZH
      REAL*4 OUTDATA(5243)
      INTEGER*4 AEROSIZE
*
*    THE FOLLOWING TYPING STATEMENTS DIMENSION THE DATA TABLE ARRAYS.
*
      INTEGER*2 DBLSIZE,RSIZE,ISIZE,HAFSIZE,BYTSIZE,SIZE,PLACE
      INTEGER*2 ELEVPT(2),DELPT(2),DERPT(2),ALPHAPT(2),MACHPT(2),
     * BETAPT(2),DSBPT(2)
      INTEGER*8 LIST(434)
*
*    THE FOLLOWING ARRAYS FORM THE I/O TABLES.
*    DATAOUT AND DATAIN FORM THE 44 CHANNEL DATA OUTPUT AND 32 CHANNEL
*    DATA INPUT TABLES, RESPECTIVELY.
*    SCALOUT AND SCALIN ARE THE CORRESPONDING OUTPUT AND INPUT SCALE
*    FACTORS, RESPECTIVELY.
      INTEGER*4 COUNT
      INTEGER*2 DATAOUT(44),DATAIN(32)
      REAL*4 SCALOUT(44),SCALIN(32)
*
*
      REAL*4      CCOT(63)
      REAL*4      DCCET(378)
      REAL*4      CCDBF1T(63)
      REAL*4      CCDBF2T(63)
      REAL*4      CCDSB1T(63)
      REAL*4      CCDSB2T(63)
      REAL*4      CLBTABL(252)
      REAL*4      CLBDE1T(63)
      REAL*4      CLBDE2T(63)
      REAL*4      CLBDSB1T(63)
      REAL*4      CLBDSB2T(63)
      REAL*4      CLBDSB3T(63)
      REAL*4      CLDAT(378)
      REAL*4      CLDRT(252)
      REAL*4      CLPT(63)
      REAL*4      CLRT(63)
      REAL*4      CMOT(63)
      REAL*4      DCMET(378)
      REAL*4      CMDBF1T(63)
      REAL*4      CMDBF2T(63)
      REAL*4      CMDSB1T(63)
      REAL*4      CMDSB2T(63)
      REAL*4      CMQT(252)
      REAL*4      CNBTABL(252)
      REAL*4      CNBDE1T(63)
      REAL*4      CNBDE2T(63)
      REAL*4      CNBDSB1T(63)
      REAL*4      CNBDSB2T(63)
      REAL*4      CNBDSB3T(63)
      REAL*4      CNDAT(378)
      REAL*4      CNDRT(252)
      REAL*4      CNDRBT(63)
      REAL*4      CNPT(63)
      REAL*4      CNRT(63)
      REAL*4      CNOT(63)
      REAL*4      CNDET(63)
      REAL*4      CNDBF1T(63)
```

```
      REAL*4       CNDBF2T(63)
      REAL*4       CNDSB1T(63)
      REAL*4       CNDSB2T(63)
      REAL*4       CYBT(63)
      REAL*4       CYDAT(378)
      REAL*4       CYDRT(63)
      REAL*4       CYDRSB1T(7)
      REAL*4       CYDRSB2T(7)
*
      REAL*4 SNDTABL(31),SNDARG(31),ZETABL(41),ZETARG(41)
      REAL*4 WINDARG(20),NWTABL(20),EWTABL(20)
      REAL*4 ELEVT(8),DELT(2),DERT(2),ALPHAT(11),MACHT(9),BETAT(6),
     * DSBT(6)
*
*
      EQUIVALENCE (INBIT,DISCRTIN)
      EQUIVALENCE (SNDARG(1),ZETARG(1))
      EQUIVALENCE (ELEVT(2),ELEVPT(1)),(DELT(2),DELPT(1)),(DERT(2),
     *DERPT(1)),(ALPHAT(2),ALPHAPT(1)),(MACHT(2),MACHPT(1)),(BETAT(2)
     *,BETAPT(1)),(DSBT(2),DSBPT(1))
*
      EQUIVALENCE
     * (OUTDATA(1),CCOT(1))
     *,(OUTDATA(64),DCCET(1))
     *,(OUTDATA(442),CCDBF1T(1))
     *,(OUTDATA(505),CCDBF2T(1))
     *,(OUTDATA(568),CCDSB1T(1))
     *,(OUTDATA(631),CCDSB2T(1))
     *,(OUTDATA(694),CLBTABL(1))
     *,(OUTDATA(946),CLBDE1T(1))
     *,(OUTDATA(1009),CLBDE2T(1))
     *,(OUTDATA(1072),CLBDSB1T(1))
     *,(OUTDATA(1135),CLBDSB2T(1))
     *,(OUTDATA(1198),CLBDSB3T(1))
     *,(OUTDATA(1261),CLDAT(1))
     *,(OUTDATA(1639),CLDRT(1))
     *,(OUTDATA(1891),CLPT (1))
     *,(OUTDATA(1954),CLRT (1))
     *,(OUTDATA(2017),CMOT (1))
     *,(OUTDATA(2080),DCMET(1))
     *,(OUTDATA(2458),CMDBF1T(1))
     *,(OUTDATA(2521),CMDBF2T(1))
     *,(OUTDATA(2584),CMDSB1T(1))
     *,(OUTDATA(2647),CMDSB2T(1))
     *,(OUTDATA(2710),CMQT(1))
     *,(OUTDATA(2962),CNBTABL(1))
     *,(OUTDATA(3214),CNBDE1T(1))
     *,(OUTDATA(3277),CNBDE2T(1))
     *,(OUTDATA(3340),CNBDSB1T(1))
     *,(OUTDATA(3403),CNBDSB2T(1))
     *,(OUTDATA(3466),CNBDSB3T(1))
     *,(OUTDATA(3529),CNDAT(1))
     *,(OUTDATA(3907),CNDRT(1))
     *,(OUTDATA(4159),CNDRBT(1))
     *,(OUTDATA(4222),CNPT(1))
     *,(OUTDATA(4285),CNRT(1))
     *,(OUTDATA(4348),CNOT(1))
     *,(OUTDATA(4411),CNDET(1))
     *,(OUTDATA(4474),CNDBF1T(1))
```

```
     *,(OUTDATA(4537),CNDBF2T(1))
     *,(OUTDATA(4600),CNDSB1T(1))
     *,(OUTDATA(4663),CNDSB2T(1))
     *,(OUTDATA(4726),CYBT(1))
     *,(OUTDATA(4789),CYDAT(1))
     *,(OUTDATA(5167),CYDRT(1))
     *,(OUTDATA(5230),CYDRSB1T(1))
     *,(OUTDATA(5237),CYDRSB2T(1))
      EQUIVALENCE
     *   (DATADBL(1),DELR)
     * ,(DATADBL(2),DELRDOT)
     * ,(DATADBL(3),GRAVITY)
     * ,(DATADBL(4),LATDOT)
     * ,(DATADBL(5),LNGDOT)
*   THERE IS A HOLE AT DATADBL(6)
     * ,(DATADBL(7),PSIHDOT)
     * ,(DATADBL(8),RUH)
     * ,(DATADBL(9),RUHDOT)
     * ,(DATADBL(10),UH)
     * ,(DATADBL(11),WH)
     * ,(DATADBL(12),XEARTH)
     * ,(DATADBL(13),XH)
     * ,(DATADBL(14),YEARTH)
     * ,(DATADBL(15),YH)
     * ,(DATADBL(16),ZEARTH)
     * ,(DATADBL(17),ZH)
*
     *,(DATAWRDR(1),CALPHA0)
     *,(DATAWRDR(2),AX)
     *,(DATAWRDR(3),AY)
     *,(DATAWRDR(4),AZ)
     *,(DATAWRDR(5),CALPHA)
     *,(DATAWRDR(6),ALTITUDE)
     *,(DATAWRDR(7),B)
     *,(DATAWRDR(8),CBETA0)
     *,(DATAWRDR(9),BETA)
     *,(DATAWRDR(10),BRAK)
     *,(DATAWRDR(11),C)
     *,(DATAWRDR(12),CC)
     *,(DATAWRDR(13),CCBF)
     *,(DATAWRDR(14),CCSB)
     *,(DATAWRDR(15),CD)
     *,(DATAWRDR(16),CL)
     *,(DATAWRDR(17),CLBT)
     *,(DATAWRDR(18),CM)
     *,(DATAWRDR(19),CMBF)
     *,(DATAWRDR(20),CMSB)
     *,(DATAWRDR(21),CN)
     *,(DATAWRDR(22),CNBF)
     *,(DATAWRDR(23),CNSB)
     *,(DATAWRDR(24),CNBT)
     *,(DATAWRDR(25),CY)
     *,(DATAWRDR(26),CLBDE)
     *,(DATAWRDR(27),CLBSB)
     *,(DATAWRDR(28),CNBDE)
     *,(DATAWRDR(29),RUNTIME)
     *,(DATAWRDR(30),TIME)
     *,(DATAWRDR(31),CNBSB)
     *,(DATAWRDR(32),CYDRSB)
```

```
*,(DATAWRDR(33),CROSSRNG)
*,(DATAWRDR(34),DAIL)
*,(DATAWRDR(35),DEL)
*,(DATAWRDR(36),ALTO)
*,(DATAWRDR(37),PRHCSOP)
*,(DATAWRDR(38),DER)
*,(DATAWRDR(39),DRIC)
*,(DATAWRDR(40),DRUD)
*,(DATAWRDR(41),DBFIC)
*,(DATAWRDR(42),DELIC)
*,(DATAWRDR(43),DACM)
*,(DATAWRDR(44),DELRIC)
*,(DATAWRDR(45),DELTA)
*,(DATAWRDR(46),DERIC)
*,(DATAWRDR(47),DOWNRNG)
*,(DATAWRDR(48),DSBIC)
*,(DATAWRDR(49),ELEV)
*,(DATAWRDR(50),EWIND)
*,(DATAWRDR(51),FLAP)
*,(DATAWRDR(52),G0)
*,(DATAWRDR(53),GEAR)
*,(DATAWRDR(54),DPJET)
*,(DATAWRDR(55),DRJET)
*,(DATAWRDR(56),HDOT)
*,(DATAWRDR(57),IXMY)
*,(DATAWRDR(58),GEOALT)
*,(DATAWRDR(59),IXX)
*,(DATAWRDR(60),IXY)
*,(DATAWRDR(61),IXZ)
*,(DATAWRDR(62),IYMZ)
*,(DATAWRDR(63),IYOY)
*,(DATAWRDR(64),IYY)
*,(DATAWRDR(65),IZMX)
*,(DATAWRDR(66),IZOZ)
*,(DATAWRDR(67),IZZ)
*,(DATAWRDR(68),IFUNC)
*,(DATAWRDR(69),J2)
*,(DATAWRDR(70),KCAS)
*,(DATAWRDR(71),KEAS)
*,(DATAWRDR(72),RRHCSOP )
*,(DATAWRDR(73),KNOTCON)
*,(DATAWRDR(74),RPTASOP)
*,(DATAWRDR(75),CLAT)
*,(DATAWRDR(76),CLATO)
*,(DATAWRDR(77),CLNG)
*,(DATAWRDR(78),CLNGO)
*,(DATAWRDR(79),UZCMD)
*,(DATAWRDR(80),UXCMD)
*,(DATAWRDR(81),LBODY)
*,(DATAWRDR(82),LENGTH)
*,(DATAWRDR(83),UYCMD)
*,(DATAWRDR(84),MACH)
*,(DATAWRDR(85),MASS)
*,(DATAWRDR(86),CMU)
*,(DATAWRDR(87),MBODY)
*,(DATAWRDR(88),NBODY)
*,(DATAWRDR(89),NWIND)
*,(DATAWRDR(90),OBLATE)
*,(DATAWRDR(91),OMEGA)
```

```
      *,(DATAWRDR(92),PBIC)
      *,(DATAWRDR(93),PDOT)
      *,(DATAWRDR(94),CPHI)
      *,(DATAWRDR(95),CPHI0)
      *,(DATAWRDR(96),CPSI)
      *,(DATAWRDR(97),CPSI0)
      *,(DATAWRDR(98),CPSIH)
      *,(DATAWRDR(99),CPSIR)
      *,(DATAWRDR(100),PBODY)
      *,(DATAWRDR(101),PHIDOT)
      *,(DATAWRDR(102),PE)
      *,(DATAWRDR(103),QBAR)
      *,(DATAWRDR(104),QBIC)
      *,(DATAWRDR(105),QDOT)
      *,(DATAWRDR(106),QBODY)
      *,(DATAWRDR(107),R0)
      *,(DATAWRDR(108),RBIC)
      *,(DATAWRDR(109),RDOT)
      *,(DATAWRDR(110),RHO)
      *,(DATAWRDR(111),RHO0)
      *,(DATAWRDR(112),SBHP)
      *,(DATAWRDR(113),RADIUS)
      *,(DATAWRDR(114),RBODY)
*    THERE IS A HOLE AT DATAWRDR(115)
      *,(DATAWRDR(116),RUHIC)
      *,(DATAWRDR(117),S)
      *,(DATAWRDR(118),CPSIH0)
      *,(DATAWRDR(119),SOUND)
      *,(DATAWRDR(120),CTHETA0)
      *,(DATAWRDR(121),CTHETA)
      *,(DATAWRDR(122),THETADOT)
      *,(DATAWRDR(123),UAIR)
      *,(DATAWRDR(124),UYGJET)
      *,(DATAWRDR(125),UBODY)
      *,(DATAWRDR(126),UEARTH)
      *,(DATAWRDR(127),VAIR)
      *,(DATAWRDR(128),VBODY)
      *,(DATAWRDR(129),VEARTH)
      *,(DATAWRDR(130),VELIC)
      *,(DATAWRDR(131),VELTRUE)
      *,(DATAWRDR(132),PITCHPAN)
      *,(DATAWRDR(133),WHIC)
      *,(DATAWRDR(134),WBODY)
      *,(DATAWRDR(135),WEARTH)
      *,(DATAWRDR(136),WEIGHT)
      *,(DATAWRDR(137),WHDOT)
      *,(DATAWRDR(138),XBAR)
      *,(DATAWRDR(139),XBRX)
      *,(DATAWRDR(140),PITCHSTK)
      *,(DATAWRDR(141),XBARNOM)
      *,(DATAWRDR(142),XBODY)
      *,(DATAWRDR(143),ROLLPAN)
      *,(DATAWRDR(144),YBAR)
      *,(DATAWRDR(145),YBRX)
      *,(DATAWRDR(146),ROLLSTK)
      *,(DATAWRDR(147),YBODY)
      *,(DATAWRDR(148),ZBAR)
      *,(DATAWRDR(149),ZBRX)
      *,(DATAWRDR(150),ZETA)
```

```
*,(DATAWRDR(151),YAWTRIM)
*,(DATAWRDR(152),ZBARNOM)
*,(DATAWRDR(153),ZBODY)
*,(DATAWRDR(154),FLAPCMD)
*,(DATAWRDR(155),CCO)
*,(DATAWRDR(156),CCE)
*,(DATAWRDR(157),CCDBF1)
*,(DATAWRDR(158),CCDBF2)
*,(DATAWRDR(159),CCDSB1)
*,(DATAWRDR(160),CCDSB2)
*,(DATAWRDR(161),CLB)
*,(DATAWRDR(162),CLBDE1)
*,(DATAWRDR(163),CLBDE2)
*,(DATAWRDR(164),CLBDSB1)
*,(DATAWRDR(165),CLBDSB2)
*,(DATAWRDR(166),CLBDSB3)
*,(DATAWRDR(167),CLDA)
*,(DATAWRDR(168),CLDR)
*,(DATAWRDR(169),CLP)
*,(DATAWRDR(170),CLR)
*,(DATAWRDR(171),CMO)
*,(DATAWRDR(172),CME)
*,(DATAWRDR(173),CMDBF1)
*,(DATAWRDR(174),CMDBF2)
*,(DATAWRDR(175),CMDSB1)
*,(DATAWRDR(176),CMDSB2)
*,(DATAWRDR(177),CMQ)
*,(DATAWRDR(178),CNB)
*,(DATAWRDR(179),CNBDE1)
*,(DATAWRDR(180),CNBDE2)
*,(DATAWRDR(181),CNBDSB1)
*,(DATAWRDR(182),CNBDSB2)
*,(DATAWRDR(183),CNBDSB3)
*,(DATAWRDR(184),CNDA)
*,(DATAWRDR(185),CNDR)
*,(DATAWRDR(186),CNDRB)
*,(DATAWRDR(187),CNP)
*,(DATAWRDR(188),CNR)
*,(DATAWRDR(189),CNO)
*,(DATAWRDR(190),CNDE)
*,(DATAWRDR(191),CNDBF1)
*,(DATAWRDR(192),CNDBF2)
*,(DATAWRDR(193),CNDSB1)
*,(DATAWRDR(194),CNDSB2)
*,(DATAWRDR(195),CYB)
*,(DATAWRDR(196),CYDA)
*,(DATAWRDR(197),CYDR)
*,(DATAWRDR(198),CYDRDSB1)
*,(DATAWRDR(199),CYDRDSB2)
*
*  ,(DATABYT(1),FCSDA)
*  ,(DATABYT(2),FCSDE)
*  ,(DATABYT(3),FCSDF)
*  ,(DATABYT(4),FCSDR)
*  ,(DATABYT(5),FCSSB)
*  ,(DATABYT(6),FIVDEG)
*  ,(DATABYT(7),TREDEG)
*  ,(DATABYT(8),WIND)
*  ,(DATABYT(9),PUTSET)
```

```
* ,(DATABYT(10),HOLD)
* ,(DATABYT(11),IC)
* ,(DATABYT(12),ABORT)
* ,(DATABYT(13),TURNCORD)
* ,(DATABYT(14),AUTOBF)
* ,(DATABYT(15),DEPULSE)
* ,(DATABYT(16),DAPULSE)
* ,(DATABYT(17),DRPULSE)
*,(ANGLE(1),ALPHA)
*,(ANGLE(2),PHI)
*,(ANGLE(3),THETA)
*
  DATA PI/3.141592654/
  DATA AEROSIZE/5243/
*
  DATA SCALOUT/.8727,3.491,.8727,.8727,1.745,.4363,128.7,64.35,
 * 64.35,1.745,25.0,2048.0,32768.0,16.0,3.142,3.142,3.142,
 * 102400.0,1024.0,1024.0,512.0,10.24,0.0,0.0,3.142,3.142,
 * 35.0,35.0,35.0,400000.0,35.0,30.0,90.0,25.0,4.096,9*0.0/
  DATA SCALIN/25.0,20.0,30.0,90.0,1.0,1.0,1.0,1.0,1.0,1.0,
 * 1.0,21*0.0/
*
  DATA ALPHAT/0.0,0.0,0.0,0.0893,0.1745,0.2618,0.3491,0.4363,
 *0.5236,0.6981,0.8727/
  DATA MACHT/0.0,0.0,1.5,2.0,3.0,4.0,5.0,8.0,10.0/
  DATA ELEVT/0.0,0.0,-35.0,-20.0,-10.0,0.0,10.0,20.0/
  DATA DSBT/0.0,0.0,0.0,25.0,55.0,87.2/
  DATA BETAT/0.0,0.0,0.0,2.0,5.0,10.0/
  DATA ALPHAPT/1,9/
  DATA MACHPT/1,7/
  DATA ELEVPT/1,6/
  DATA DELPT/1,6/
  DATA DERPT/1,6/
  DATA DSBPT/1,4/
  DATA BETAPT/1,4/
  DATA ZETARG/0.,10000.,20000.,30000.,40000.,50000.,60000.,70000.,
 1 80000.,90000.,100000.,110000.,120000.,130000.,140000.,150000.,
 1 160000.,170000.,180000.,190000.,200000.,210000.,220000.,230000.,
 1 240000.,250000.,260000.,270000.,280000.,290000.,300000.,310000.,
 * 320000.,330000.,340000.,350000.,360000.,370000.,380000.,390000.,
 * 400000./
*
  DATA SNDTABL/1116.,1077.,1037.,995.,968.,968.,968.,971.,978.,985.,
 1 991.,1004.,1022.,1040.,1058.,1075.,1082.,1082.,1071.,1058.,1046.,
 1 1020.,994.,967.,939.,911.,884.,884.,884.,884.,904./
  DATA ZETABL/.292330,.303149,.314793,.327719,.350427,.376466
 * ,.393832,.407019,.417173,.424332,.429408,.433938,.437363,.438992
 * ,.439288,.438588,.436120,.433098,.429438,.426540,.424377,.421935
 * ,.420577,.420275,.420998,.422748,.425734,.431314,.436493,.441318
 * ,.446949,.451434,.454810,.457721,.459620,.460466,.461186,.461258
 * ,.460134,.458847,.457300/
*
  DATA WINDARG/0.,5000.,10000.,15000.,20000.,25000.,30000.,35000.,
 1 40000.,45000.,50000.,55000.,60000.,65000.,70000.,75000.,80000.,
 1 85000.,90000.,95000./
*
  DATA DBLSIZE,RSIZE,ISIZE,HAFSIZE,BYTSIZE,SIZE/17,400,0,0,17,434/
*
*   LIST CONTAINS THE ASCII DOUBLEWORDS WHICH DEFINE VARIABLE NAMES TO
```

```
*    EASE AND DATASTORE.
*
       DATA LIST/
*    REAL DOUBLEWORD VARIABLES ARE DEFINED HERE.
      *   'DELR'
      * ,'DELRDOT'
      * ,'GRAVITY'
      * ,'LATDOT'
      * ,'LNGDOT'
*    THIS HOLE CORRESPONDS TO DATADBL(6)
      * ,' '
      * ,'PSIHDOT'
      * ,'RUH'
      * ,'RUHDOT'
      * ,'UH'
      * ,'WH'
      * ,'XEARTH'
      * ,'XH'
      * ,'YEARTH'
      * ,'YH'
      * ,'ZEARTH'
      * ,'ZH'
*
*    THE REAL WORD VARIABLES ARE DEFINED HERE.
      * ,'ALPHAO'
      * ,'AX'
      * ,'AY'
      * ,'AZ'
      * ,'ALPHA'
      * ,'ALTITUDE'
      * ,'B'
      * ,'BETAO'
      * ,'BETA'
      * ,'BRAK'
      * ,'C'
      * ,'CC'
      * ,'CCBF'
      * ,'CCSB'
      * ,'CD'
      * ,'CL'
      * ,'CLBT'
      * ,'CM'
      * ,'CMBF'
      * ,'CMSB'
      * ,'CN'
      * ,'CNBF'
      * ,'CNSB'
      * ,'CNBT'
      * ,'CY'
      * ,'CLBDE'
      * ,'CLBSB'
      * ,'CNBDE'
      * ,'RUNTIME'
      * ,'TIME'
      * ,'CNBSB'
      * ,'CYDRSB'
      * ,'CROSSRNG'
      * ,'DAIL'
      * ,'DEL'
```

```
* ,'ALTO'
* ,'PRHCSOP '
* ,'DER'
* ,'DRIC'
* ,'DRUD'
* ,'DBFIC'
* ,'DELIC'
* ,'DACM'
* ,'DELRIC'
* ,'DELTA'
* ,'DERIC'
* ,'DOWNRNG'
* ,'DSBIC'
* ,'ELEV'
* ,'EWIND'
* ,'FLAP'
* ,'GO'
* ,'GEAR'
* ,'DPJET'
* ,'DRJET'
* ,'HDOT'
* ,'IXMY'
* ,'GEOALT'
* ,'IXX'
* ,'IXY'
* ,'IXZ'
* ,'IYMZ'
* ,'IYOY'
* ,'IYY'
* ,'IZMX'
* ,'IZOZ'
* ,'IZZ'
* ,'IFUNC'
* ,'J2'
* ,'KCAS'
* ,'KEAS'
* ,'RRHCSOP '
* ,'KNOTCON'
* ,'RPIASOP '
* ,'LAT'
* ,'LATO'
* ,'LNG'
* ,'LNGO'
* ,'UZCMD'
* ,'UXCMD'
* ,'LBODY'
* ,'LENGTH'
* ,'UYCMD'
* ,'MACH'
* ,'MASS'
* ,'MU'
* ,'MBODY'
* ,'NBODY'
* ,'NWIND'
* ,'OBLATE'
* ,'OMEGA'
* ,'PBIC'
* ,'PDOT'
* ,'PHI'
```

```
      * ,'PHIO'
      * ,'PSI'
      * ,'PSIO'
      * ,'PSIH'
      * ,'PSIR'
      * ,'PBODY'
      * ,'PHIDOT'
      * ,'PE'
      * ,'QBAR'
      * ,'QBIC'
      * ,'QDOT'
      * ,'QBODY'
      * ,'RO'
      * ,'RBIC'
      * ,'RDOT'
      * ,'RHO'
      * ,'RHOO'
      * ,'SBHP    '
      * ,'RADIUS'
      * ,'RBODY'
*     THIS HOLE CORRESPONDS TO DATAWRDR(115)
      * ,' '
      * ,'RUHIC'
      * ,'S'
      * ,'PSIHO'
      * ,'SOUND'
      * ,'THETAO'
      * ,'THETA'
      * ,'THETADOT'
      * ,'UAIR'
      * ,'UYGJET'
      * ,'UBODY'
      * ,'UEARTH'
      * ,'VAIR'
      * ,'VBODY'
      * ,'VEARTH'
      * ,'VELIC'
      * ,'VELTRUE'
      * ,'PITCHPAN'
      * ,'WHIC'
      * ,'WBODY'
      * ,'WEARTH'
      * ,'WEIGHT'
      * ,'WHDOT'
      * ,'XBAR'
      * ,'XBRX'
      * ,'PITCHSTK'
      * ,'XBARNOM'
      * ,'XBODY'
      * ,'ROLLPAN'
      * ,'YBAR'
      * ,'YBRX'
      * ,'ROLLSTK'
      * ,'YBODY'
      * ,'ZBAR'
      * ,'ZBRX'
      * ,'ZETA'
      * ,'YAWTRIM'
      * ,'ZBARNOM'
```

```
     * ,'ZBODY'
     * ,'FLAPCMD'
*
*   THIS POINT,REFERENCED BY THE DATASTORE SUBROUTINE,IS LIST(172)
     * ,'CCO'
     * ,'CCE'
     * ,'CCDBF1'
     * ,'CCDBF2'
     * ,'CCDSB1'
     * ,'CCDSB2'
     * ,'CLB'
     * ,'CLBDE1'
     * ,'CLBDE2'
     * ,'CLBDSB1'
     * ,'CLBDSB2'
     * ,'CLBDSB3'
     * ,'CLDA'
     * ,'CLDR'
     * ,'CLP'
     * ,'CLR'
     * ,'CMO'
     * ,'CME'
     * ,'CMDBF1'
     * ,'CMDBF2'
     * ,'CMDSB1'
     * ,'CMDSB2'
     * ,'CMQ'
     * ,'CNB'
     * ,'CNBDE1'
     * ,'CNBDE2'
     * ,'CNBDSB1'
     * ,'CNBDSB2'
     * ,'CNBDSB3'
     * ,'CNDA'
     * ,'CNDR'
     * ,'CNDRB'
     * ,'CNP'
     * ,'CNR'
     * ,'CNO'
     * ,'CNDE'
     * ,'CNDBF1'
     * ,'CNDBF2'
     * ,'CNDSB1'
     * ,'CNDSB2'
     * ,'CYB'
     * ,'CYDA'
     * ,'CYDR'
     * ,'CYDRDSB1'
     * ,'CYDRDSB2'
     * ,'DEMAN'
     * ,'ESHAPE'
     * ,'GPS'
     * ,'DEMS'
     * ,'Q'
     * ,'PROD26'
     * ,' '
     * ,'GJET'
     * ,' '
     * ,'ETRIMIN'
```

```
* ,'ETRIM'
* ,'ELFBKIN'
* ,'ELFBK'
* ,'DETMPAN'
* ,'DSBPC'
* ,'DSBXTR'
* ,'DSBXTRS'
* ,'DETDSB'
* ,'DETRIM'
* ,'GDSB'
* ,'GTRE'
* ,'DETMRHC'
* ,'DETR'
* ,'DEMSP'
* ,'RTPHI'
* ,'RTANPHI'
* ,'DJRTP'
* ,'BCSL'
* ,'DCSL'
* ,'DQCT'
* ,'DECMD'
* ,'QFFIL'
* ,'DCSQ'
* ,'DCSLLO'
* ,'DCSLHI'
* ,'MIDPICK'
* ,'KPIT'
* ,'GDQ'
* ,'DQLO'
* ,'R'
* ,'DQHI'
* ,'DQHIA'
* ,'DQHIN'
* ,'NZA'
* ,'NZP'
* ,'DQLON'
* ,'DQLOA'
* ,'ALPHMIN'
* ,'ALPHMAX'
* ,'GTREDQCT'
* ,'P'
* ,' '
* ,'DATMRHC'
* ,'PROD24'
* ,'PROD25'
* ,2*' '
* ,'PROD21'
* ,'PROD22'
* ,'PROD23'
* ,'PROD27'
* ,'SUM21'
* ,' '
* ,'SUM23'
* ,'DCSP'
* ,'DACMD'
* ,'DATSUMI'
* ,'KDAMLIN'
* ,'KDAMPAR'
* ,'GRS'
```

```
*  ,'GDAC'
*  ,'GDA'
*  ,'DATSUM'
*  ,'PROD42'
*  ,'DATR'
*  ,'DRPRM'
*  ,'PSTAB'
*  ,'SUM24'
*  ,'DRPHI'
*  ,'PROD29'
*  ,'SUM22'
*  ,'GTX'
*  ,'YAWXFEED'
*  ,'PROD41'
*  ,'GP'
*  ,'GRH'
*  ,' '
*  ,' '
*  ,'SUM25'
*  ,'SUM25S'
*  ,' '
*  ,'SUM26'
*  ,'PROD43'
*  ,'RSTAB'
*  ,'BETAQ'
*  ,'BETAFILT'
*  ,'DATMPAN'
*  ,'DATP'
*  ,' '
*  ,'PROD44'
*  ,'DRT'
*  ,'DRTMSF'
*  ,'DRMS'
*  ,'DRTMS'
*  ,'PROD1'
*  ,'SUM1'
*  ,'NYP'
*  ,'NYA'
*  ,'GRAY'
*  ,'PROD2'
*  ,'PROD3'
*  ,'SUM2'
*  ,'PROD5'
*  ,'DRCPF'
*  ,'KDRC'
*  ,'GDRC'
*  ,'GNYDRM'
*  ,'DRTRR'
*  ,'PROD7'
*  ,'GTRR'
*  ,'DRTRIM'
*  ,'SUM3'
*  ,'DRCMD'
*  ,'DACMC'
*  ,'GRXFD'
*  ,'GDRE'
*  ,' '
*  ,'NR'
*  ,73*' '
```

```
*
*    THE LOGICAL BYTE VARIABLES ARE DEFINED HERE.
      * ,'FCSDA'
      * ,'FCSDE'
      * ,'FCSDF'
      * ,'FCSDR'
      * ,'FCSSB'
      * ,'FIVDEG'
      * ,'TREDEG'
      * ,'WIND'
      * ,'POTSET'
      * ,'HOLD'
      * ,'IC'
      * ,'ABORT'
      * ,'TURNCORD'
      * ,'AUTOBF'
      * ,'DEPULSE'
      * ,'DAPULSE'
      * ,'DRPULSE'/
*EXECUTIVE PORTION FOLLOWS
*THIS IS THE ENTRY POINT FOR PROGRAM ACTIVATION
**
*INITIALIZATION
*
*    THIS SECTION INITIALIZES THE INTERVAL TIMER HANDLER.
*
      INLINE
)1020    DI      43          DISABLE INTERVAL TIMER
         LW      7,428       FETCH ADDRESS OF OLD HANDLER
         STW     7,HANDLER   STORE FOR LATER USE
         LW      7,)1000     FETCH ADDRESS OF NEW HANDLER
         STW     7,428       STORE ADDRESS IN HANDLER LOC.
      ENDI
*
*
*    THE AERODYNAMIC DATA IS PLACED IN CORE HERE
*
      II=1
      JJ=1152
    3 CALL BUFFERIN(5,1,OUTDATA(II),JJ)
    4 CALL STATUS(5,TEMP1)
      IF(TEMP1.NE.2)GOTO4
      II=II+1152
      IF(II.GT.AEROSIZE)GO TO 5
      IF(II+1152.GT.AEROSIZE)JJ=AEROSIZE+1-II
      GO TO 3
    5 CONTINUE
*
      IC=.TRUE.
      POTSET=.FALSE.
      HOLD=.FALSE.
*
*    THE FOLLOWING VARIABLES ARE SET TO INDICATE A DESIRED CONDITION
*    AS INDICATED. THESE VARIABLES ARE EASE ACCESSABLE.
*
*    AUTO BODY FLAP (NOT AVAILABLE)
      AUTOBF=.FALSE.
*    TURN COORDINATION
      TURNCORD=.TRUE.
```

```
*    ORDERLY PROGRAM TERMINATION
       ABORT=.FALSE.
*    FIVE DEGREES OF FREEDOM
       FIVDEG=.TRUE.
*    THREE DEGREES OF FREEDOM
       TREDEG=.FALSE.
*    ATMOSPHERIC WIND
       WIND=.FALSE.
*    CONTROL OF VEHICLE SURFACE DEFLECTIONS....TRUE. INDICATES THAT THE
*    VALUES ARE DETERMINED BY DELIC,DERIC,DRIC,DSBIC,DBFIC.
       FCSDE=FCSDA=FCSDR=.FALSE.
       FCSDF=FCSSB=.TRUE.
*    SURFACE PULSE FOR DYNAMIC RESPONSE CHECK....TRUE. INDICATES THAT
*    THE SURFACE IS TO BE PULSED.
       DEPULSE=DAPULSE=DRPULSE=.FALSE.
*    PRESET PROGRAM RUN TIME IN SECONDS.
       RUNTIME=5.E75
*
       CALPHA0=20.0
       CBETA0=0.0
       CTHETA0=20.0
       CPHI0=0.0
       CPSI0=0.0
       CLAT0=0.0
       CLNG0=0.0
       CPSIR=26.0
       S=2690.0
       B=78.03
       C=39.56
       LENGTH=107.5
       WEIGHT=155000.0
       IXX=600000.0
       IYY=5500000.0
       IZZ=5500000.0
       IXY=0.0
       IXZ=100000.0
       XBAR=0.65
       YBAR=0.0
       ZBAR=0.2907
       XBARNOM=0.65
       ZBARNOM=0.2907
       G0=32.146546
       J2=.0010823
       KNOTCON=0.59210526
       OBLATE=70150.0
       OMEGA=.000072921159
       R0=20925738.0
       RHO0=.0023769
       DBFIC=-5.0
       DELIC=-16.33
       DERIC=-16.33
       DRIC=0.0
       DSBIC=25.0
       ALT0=99480.0
       PBIC=0.0
       QBIC=0.0
       RBIC=0.0
       VELIC=2970.0
       DELTA=.04
```

```
      HALFPI = PI/2
      TWOPI = 2*PI
*
*THIS IS THE EASE RETURN POINT
   10 IF(ABORT) GO TO 1100
*
*    A DIGITAL INPUT IS PERFORMED HERE TO INITIALIZE THE MODE CONTROL
*    PARAMETERS.
   15 CONTINUE
      INLINE
         LW        4,820003FFFF
         LEA       1,)920
         SIMW      1,256
         CD        0,428000
      ENDI
*
      MASS=WEIGHT/32.174
      RE=R0-OBLATE
      HALFDEL=DELTA/2
*
*PITS1,PITS2,PIT3 ARE TEMPORARY LOCATIONS
      PIT1=MASS*LENGTH
      XBRX=(XBARNOM-XBAR)*PIT1
      YBRX=YBAR*PIT1
      ZBRX=(ZBARNOM-ZBAR)*PIT1
      IXMY=IXX-IYY
      IYMZ=IYY-IZZ
      IZMX=IZZ-IXX
      IYOY=IXY/IYY
      IZOZ=IXZ/IZZ
      IFUNC=1/(IXX-IZOZ*IXZ-IYOY*IXY)
      SC=S*C
      SC2BY2=.5*S*C**2
      SB=S*B
      SB2BY2=.5*S*B**2
*    THE FOLLOWING SECTION CALCULATES THE INITIAL CONDITION VALUES
*    FOR THE INTEGRAL EQUATIONS. EASE DEFINES THETA,PHI,PSI,ALPHA,
*    BETA,VELTRUE AND ALTITUDE. THESE VALUES ARE PROCESSED TO OBTAIN
*    ROH,PSIH,WH,AND DELR.
*    ALL ANGLES ARE INTERNALLY DEFINED IN RADIANS. ALL COMMUNICATIONS
*    WITH CARDS AND EASE ARE WITH ANGLES DEFINED IN DEGREES. A "C"
*    PRECEEDS ALL DEGREE-DEFINED ANGLES.
      THETA0=CTHETA0/57.29577951
      PHI0=CPHI0/57.29577951
      PSI0=CPSI0/57.29577951
      ALPHA0=CALPHA0/57.29577951
      BETA0=CBETA0/57.29577951
      PSIR=CPSIR/57.29577951
      LAT0=CLAT0/57.29577951
      LNG0=CLNG0/57.29577951
      PIT1=SIN(THETA0)
      PIT2=COS(THETA0)
      PIT3=SIN(PHI0)
      PIT4=COS(PHI0)
      PIT5=SIN(PSI0)
      PIT6=COS(PSI0)
      L1=PIT2*PIT6
      L2=PIT2*PIT5
      L3=-PIT1
```

```
      M1=-PIT4*PIT5+PIT3*PIT1*PIT6
      M2=PIT4*PIT6+PIT3*PIT1*PIT5
      M3=PIT3*PIT2
      N1=PIT3*PIT5+PIT4*PIT1*PIT6
      N2=-PIT3*PIT6+PIT4*PIT1*PIT5
      N3=PIT2*PIT4
      DELRIC=ALTO-R0*(.00167828-.00167616*COS(2*LATO)-.0000211*
     * COS(4*LATO))
      RAD=R0+DELRIC
      UBODY=VELIC*COS(ALPHA0)*COS(BETA0)
      VBODY=VELIC*SIN(BETA0)
      WBODY=VELIC*SIN(ALPHA0)*COS(BETA0)
      UAIR=UBODY*L1+VBODY*M1+WBODY*N1
      VAIR=UBODY*L2+VBODY*M2+WBODY*N2
      WEARTH=UBODY*L3+VBODY*M3+WBODY*N3
      UEARTH=UAIR
      VEARTH=VAIR+RAD*COS(LATO)*OMEGA
      PSIH0=ATAN2(VEARTH,UEARTH)
      ROHIC=RAD*UEARTH/COS(PSIH0)
      WHIC=WEARTH
      CPSIH0=PSIH0*57.29577951
*
*    THIS SECTION STARTS THE INTERVAL TIMER OPERATION.
*
      COUNT=DELTA/.0000012
      INLINE
         LW       0,COUNT        LOAD FRAME TIME
         CD       127,60         START TIMER
         EI       43             ENABLE INTERVAL TIMER
      ENDI
*
*PATCHES WILL GO HERE
*END OF INITIALIZATION ROUTINE
*THIS IS THE LOOP RETURN ENTRY POINT
   47 POTSET=INBIT(1)
      IC=INBIT(2)
      HOLD=INBIT(3)
*
      IF(POTSET) GO TO 1105
      IF(HOLD) GO TO 170
      IF(IC) TIME=0
*
*    ANALOG INPUT IS STARTED HERE.
      INLINE
         LW       4,8Z0003FFFF
         LEA      1,)910
         STMW     1,256
         CD       0,4Z8000
      ENDI
*
*    THIS SECTION TRANSFERS THE INPUT VALUES CONTAINED IN THE DATA
*    HALFWORD TABLES TO THE APPROPRIATE REAL WORD VARIABLES.
      JJ=1
      INLINE
         BL       )960
      ENDI
      PRHCSUP=DATA
      JJ=2
      INLINE
```

```
         BL        )960
      ENDI
      RRHCSOP=DATA
      JJ=3
      INLINE
         BL        )960
      ENDI
      RPTASOP=DATA
      JJ=4
      INLINE
         BL        )960
      ENDI
      SBHP=DATA
      JJ=5
      INLINE
         BL        )960
      ENDI
      PITCHPAN=DATA
      JJ=6
      INLINE
         BL        )960
      ENDI
      PITCHSTK=DATA
      JJ=7
      INLINE
         BL        )960
      ENDI
      ROLLPAN=DATA
      JJ=8
      INLINE
         BL        )960
      ENDI
      ROLLSTK=DATA
      JJ=9
      INLINE
         BL        )960
      ENDI
      YAWTRIM=DATA
      JJ=10
      INLINE
         BL        )960
      ENDI
      FLAPCMD=DATA
      JJ=11
      INLINE
         BL        )960
      ENDI
      PULSE=DATA
*
      IF(IC) GO TO 49
      GO TO 170
   49 PBODY=PBIC;QBODY=QBIC;RBODY=RBIC
      THETA = THETA0 ; PHI = PHI0 ; PSI = PSI0
      WH=WHIC;DELR=DELRIC
      PSIH=PSIH0;RUH=RUHIC
      LA=LAT0;LN=LNG0
  170 CONTINUE
*
*
```

```
C   THIS STARTS THE DATA LOOK-UP.
C   FIND ARGUMENT BREAKPOINTS
      ABSBETA=ABS(BETA)
      CALL POINTF(MACH,MACHPT(2),MACHPT(1),MACHT(1),MACHT(3))
      CALL POINTF(ALPHA,ALPHAPT(2),ALPHAPT(1),ALPHAT(1),ALPHAT(3))
      CALL POINTF(ELEV,ELEVPT(2),ELEVPT(1),ELEVT(1),ELEVT(3))
      CALL POINTF(DEL,DELPT(2),DELPT(1),DELT(1),ELEVT(3))
      CALL POINTF(DER,DERPT(2),DERPT(1),DERT(1),ELEVT(3))
      CALL POINTF(BRAK,DSBPT(2),DSBPT(1),DSBT(1),DSBT(3))
      CALL POINTF(ABSBETA,BETAPT(2),BETAPT(1),BETAT(1),BETAT(3))
C   FIND DERIVATIVE VALUES
      CMO=DERIVE2(CMOT,ALPHAPT(2),MACHPT(2),ALPHAPT(1),MACHPT(1),ALPHAT(
     *1),MACHT(1))
      CNO=DERIVE2(CNOT,ALPHAPT(2))
      CNDE=DERIVE2(CNDET,ALPHAPT(2))
      CCO=DERIVE2(CCOT,ALPHAPT(2))
      CLP=DERIVE2(CLPT,ALPHAPT(2))
      CLR=DERIVE2(CLRT,ALPHAPT(2))
      CNDRB=DERIVE2(CNDRBT,ALPHAPT(2))
      CNP=DERIVE2(CNPT,ALPHAPT(2))
      CNR=DERIVE2(CNRT,ALPHAPT(2))
      CYB=DERIVE2(CYBT,ALPHAPT(2))
      CYDR=DERIVE2(CYDRT,ALPHAPT(2))
      IF (FLAP.GT.0.0) GOTO 220
      CCBF=CCDBF1=DERIVE2(CCDBF1T,ALPHAPT(2))
      CMBF=CMDBF1=DERIVE2(CMDBF1T,ALPHAPT(2))
      CNBF=CNDBF1=DERIVE2(CNDBF1T,ALPHAPT(2))
      GOTO 230
  220 CONTINUE
      CCBF=CCDBF2=DERIVE2(CCDBF2T,ALPHAPT(2))
      CMBF=CMDBF2=DERIVE2(CMDBF2T,ALPHAPT(2))
      CNBF=CNDBF2=DERIVE2(CNDBF2T,ALPHAPT(2))
  230 CONTINUE
      IF (BRAK.GT.25.0) GOTO240
      CLBSB=CLBDSB1=DERIVE2(CLBDSB1T,ALPHAPT(2))
      CNBSB=CNBDSB1=DERIVE2(CNBDSB1T,ALPHAPT(2))
      GOTO 245
  240 CONTINUE
      CLBSB=CLBDSB2=DERIVE2(CLBDSB2T,ALPHAPT(2))
      CNBSB=CNBDSB2=DERIVE2(CNBDSB2T,ALPHAPT(2))
  245 CONTINUE
      CLBDSB3=CNBDSB3=0.0
      IF (BRAK.LT.60.0) GOTO 249
      CLBDSB3=DERIVE2(CLBDSB3T,ALPHAPT(2))
      CNBDSB3=DERIVE2(CNBDSB3T,ALPHAPT(2))
  249 CONTINUE
      IF (ELEV.GT.0.0) GOTO 250
      CLBDE=CLBDE1=DERIVE2(CLBDE1T,ALPHAPT(2))
      CNBDE=CNBDE1=DERIVE2(CNBDE1T,ALPHAPT(2))
      GOTO 255
  250 CLBDE=CLBDE2=DERIVE2(CLBDE2T,ALPHAPT(2))
      CNBDE=CNBDE2=DERIVE2(CNBDE2T,ALPHAPT(2))
  255 CONTINUE
      IF (BRAK.GT.55.0) GOTO 258
      CCSB=CCDSB1=DERIVE2(CCDSB1T,ALPHAPT(2))
      CMSB=CMDSB1=DERIVE2(CMDSB1T,ALPHAPT(2))
      CNSB=CNDSB1=DERIVE2(CNDSB1T,ALPHAPT(2))
      CYDRSB=CYDRDSB1=DERIVE1(CYDRSB1T,MACHPT(2),MACHPT(1),MACHT(1))
      GOTO 259
```

```
  258 CONTINUE
      CCSB=CCDSB2=DERIVE2(CCDSB2T,ALPHAPT(2))
      CMSB=CMDSB2=DERIVE2(CMDSB2T,ALPHAPT(2))
      CNSB=CNDSB2=DERIVE2(CNDSB2T,ALPHAPT(2))
      CYDRSB=CYDRDSB2=DERIVE1(CYDRSB2T,MACHPT(2),MACHPT(1),MACHT(1))
  259 CONTINUE
      CMEL=DERIVE3(DCMET,ALPHAPT(2),MACHPT(2),DELPT(2),ALPHAPT(1),
     * MACHPT(1),DELPT(1),ALPHAT(1),MACHT(1),DELT(1))
      CCEL=DERIVE3(DCCET,ALPHAPT(2))
      CMER=DERIVE3(DCMET,ALPHAPT(2),MACHPT(2),DERPT(2),ALPHAPT(1),
     * MACHPT(1),DERPT(1),ALPHAT(1),MACHT(1),DERT(1))
      CCER=DERIVE3(DCCET,ALPHAPT(2))
      CLDA=DERIVE3(CLDAT,ALPHAPT(2),MACHPT(2),ELEVPT(2),ALPHAPT(1),
     *MACHPT(1),ELEVPT(1),ALPHAT(1),MACHT(1),ELEVT(1))
      CNDA=DERIVE3(CNDAT,ALPHAPT(2))
      CYDA=DERIVE3(CYDAT,ALPHAPT(2))
      CLB=DERIVE3(CLBTABL,ALPHAPT(2),MACHPT(2),BETAPT(2),ALPHAPT(1)
     *,MACHPT(1),BETAPT(1),ALPHAT(1),MACHT(1),BETAT(1))
      CNB=DERIVE3(CNBTABL,ALPHAPT(2))
      CMQ=DERIVE3(CMQT,ALPHAPT(2),MACHPT(2),DSBPT(2),ALPHAPT(1),
     *MACHPT(1),DSBPT(1),ALPHAT(1),MACHT(1),DSBT(1))
      CLDR=DERIVE3(CLDRT,ALPHAPT(2))
      CNDR=DERIVE3(CNDRT,ALPHAPT(2))
      CME=(CMEL+CMER)/2.0
      CCE=(CCEL+CCER)/2.0
C   THIS CONCLUDES THE DATA LOOK-UP.
*
*CALCULATION OF DERIVATIVE COEFFICIENTS
*CL AND LBODY
*CALCULATION OF CLBT
      CLBT=CLB+CLBDE*ELEV+CLBSB*(BRAK-25)+CLBDSB3*(BRAK-60)
*CALCULATION OF LBODY
      LBODY=QBAR*((CLBT*BETA+CLDA*DAIL+CLDR*DRUD)*SB
     * +SB2BY2*(CLP*PBODY+CLR*RBODY)/VELTRUE)-YBRX*ZBODY+ZBRX*YBODY
     * -UZCMD*6125
*CM AND MBODY
*CALCULATION OF CM
      CM=CMO+CME+CMBF*FLAP+CMSB*(BRAK-55)
*CALCULATION OF MBODY
      MBODY=QBAR*(CM*SC+CMQ*SC2BY2*QBODY/VELTRUE)-ZBRX*XBODY+XBRX*ZBODY
*CN AND NBODY
*CALCULATION OF CNBT
      CNBT=CNB+CNBDE*ELEV+CNBSB*(BRAK-25)+CNBDSB3*(BRAK-60)
*CALCULATION OF NBODY
      NBODY=QBAR*((CNBT*BETA+CNDA*DAIL+(CNDR+CNDRB*ABSBETA)*DRUD)*SB+
     * (CNR*RBODY+CNP*PBODY)*SB2BY2/VELTRUE)-XBRX*YBODY+YBRX*XBODY
     * +UZCMD*33469
*CALCULATION OF CC
      CC=CCO+CCE+CCBF*FLAP+CCSB*(BRAK-55)
*CALCULATION OF CY
      CY=CYB*BETA+CYDA*DAIL+(CYDR+CYDRSB*(BRAK-55))*DRUD
* CALCULATION OF CN (NORMAL FORCE COEFFICIENT)
      CN=CNO+CNDE*ELEV+CNBF*FLAP+CNSB*(BRAK-55)
* CALCULATION OF CL (LIFT COEFFICIENT)
      CL = CN*COSALF -CC*SINALF
* CALCULATION OF CD
      CD = CC*COSALF + CN*SINALF
*
*
```

```
*THE FOLLOWING IS A FOURTH ORDER RUNGE KUTTA INTEGRATION ROUTINE
*FOR FORMATION OF PBODY,QBODY,AND RBODY
*THIS SECTION USES FUNCTION TYPE SUBPROGRAMS FOR CALCULATION OF
*DERIVATIVE TERMS
*DELTA=FRAME TIME IN SECONDS
*
      PDOT=PDOTT(PBODY,QBODY,RBODY)
      K0=DELTA*PDOT
      QDOT=QDOTT(PBODY,QBODY,RBODY,PDOT)
      L0=DELTA*QDOT
      RDOT=RDOTT(PBODY,QBODY,RBODY,PDOT)
      M0=DELTA*RDOT
*
      PIT1=PDOTT(PBODY+.5*K0,QBODY+.5*L0,RBODY+.5*M0)
      K1=DELTA*PIT1
      L1=DELTA*QDOTT(PBODY+.5*K0,QBODY+.5*L0,RBODY+.5*M0,PIT1)
      M1=DELTA*RDOTT(PBODY+.5*K0,QBODY+.5*L0,RBODY+.5*M0,PIT1)
*
      PIT1=PDOTT(PBODY+.5*K1,QBODY+.5*L1,RBODY+.5*M1)
      K2=DELTA*PIT1
      L2=DELTA*QDOTT(PBODY+.5*K1,QBODY+.5*L1,RBODY+.5*M1,PIT1)
      M2=DELTA*RDOTT(PBODY+.5*K1,QBODY+.5*L1,RBODY+.5*M1,PIT1)
*
      PIT1=PDOTT(PBODY+K2,QBODY+L2,RBODY+M2)
      K3=DELTA*PIT1
      L3=DELTA*QDOTT(PBODY+K2,QBODY+L2,RBODY+M2,PIT1)
      M3=DELTA*RDOTT(PBODY+K2,QBODY+L2,RBODY+M2,PIT1)
*
*STOP INTEGRATION IF IC OR HOLD MODES
      IF(IC.OR.HOLD) GO TO 210
*
      PBODY=PBODY+K0/6+K1/3+K2/3+K3/6
      QBODY=QBODY+L0/6+L1/3+L2/3+L3/6
      RBODY=RBODY+M0/6+M1/3+M2/3+M3/6
*
 210  CONTINUE
*
*THIS SECTION USES THE FOURTH ORDER RUNGE KUTTA TECHNIQUE TO GENERATE
*THE EULER ANGLES THETA,PHI,AND PSI
*THE RESULTANT VALUES FOR THETA,PHI,ANDPSI ARE IN RADIANS
*THE SIN AND COS VALUES FOR THE EULER ANGLES ARE ALSO CALCULATED
      SINTHETA=SIN(THETA)
      COSTHETA=COS(THETA)
      SINPHI=SIN(PHI)
      COSPHI=COS(PHI)
      SINPSI=SIN(PSI)
      COSPSI=COS(PSI)
*
      THETADOT=QBODY*COSPHI-RBODY*SINPHI
      K00=DELTA*THETADOT
      PSIDOT=(RBODY*COSPHI+QBODY*SINPHI)/COSTHETA
      L00=DELTA*PSIDOT
      PHIDOT=PBODY+PSIDOT*SINTHETA
      M00=DELTA*PHIDOT
*
      K01=DELTA*THETDOTT(PHI+.5*M00)
      PIT1=PSIDOTT(PHI+.5*M00,THETA+.5*K00)
      L01=DELTA*PIT1
      M01=DELTA*PHIDOTT(PHI+.5*M00,THETA+.5*K00,PIT1)
```

```
*
      K02=DELTA*THETDOTT(PHI+.5*M01)
      PIT1=PSIDOTT(PHI+.5*M01,THETA+.5*K01)
      L02=DELTA*PIT1
      M02=DELTA*PHIDOTT(PHI+.5*M01,THETA+.5*K01,PIT1)
*
      K03=DELTA*THETDOTT(PHI+M02)
      PIT1=PSIDOTT(PHI+M02,THETA+K02)
      L03=DELTA*PIT1
      M03=DELTA*PHIDOTT(PHI+M02,THETA+K02,PIT1)
*
*INTEGRATION IS NOT PERFORMED IN THE IC OR HOLD MODES
      IF(IC.OR.HOLD) GO TO 270
*THREE DEGREES OF FREEDOM MODE REQUIRES PHIDOT AND PSI DOT =0
      IF(TREDEG) L00=L01=L02=L03=M00=M01=M02=M03=0
*
      THETA=THETA+K00/6+K01/3+K02/3+K03/6
      PHI=PHI+M00/6+M01/3+M02/3+M03/6
      PSI=PSI+L00/6+L01/3+L02/3+L03/6
*
 270  CONTINUE
*
*CALCULATION OF GRAVITY,INCLUDING EARTHS OBLATENESS
      MU=HALFPI-LA
      COSMU=DCOS(MU)
      SINMU=DSIN(MU)
*THIS TERM,SHARE,IS USED BOTH HERE AND IN THE CALCULATION OF XEARTH
*J2,GCON1,ANDGCON2 ARE CONSTANTS
      SHARE=-3*J2*G0*(R0/RAD)**4
      GRAVITY=G0*(R0/RAD)**2+.5*SHARE*(3*COSMU**2-1)-UH**2/RAD
*GRAVITY HAS UNITS FT/SEC**2
*
* BODY AXIS ACCELERATION CALCULATIONS
      DPIT1=QBAR/MASS
*TEST FOR DESIRED FIVE DEGREES OF FREEDOM
      IF (FIVDEG) GO TO 290
      XB=-DPIT1*CC*S
      GO TO 320
  290 DPIT2=(WB*ZB+VB*YB)/UB
      DPIT3=SINTHETA-COSTHETA*(WB*COSPHI+VB*SINPHI)/UB
      XB=GRAVITY*DPIT3-DPIT2
  320 AX=XB
      XBODY=XB
      YB=DPIT1*CY*S
      AY=YB
      YBODY=YB
      ZB=-DPIT1*CN*S
      AZ=ZB
      ZBODY=ZB
* EARTH AND H-FRAME ACCELERATIONS
      DPIT1=ZB*COSPHI+YB*SINPHI
      DPIT2=-ZB*SINPHI+YB*COSPHI
      DPIT3=XB*COSTHETA+DPIT1*SINTHETA
      XEARTH=-DPIT2*SINPSI+DPIT3*COSPSI+SHARE*COSMU*SINMU
      YEARTH=DPIT2*COSPSI+DPIT3*SINPSI
      ZEARTH=-XB*SINTHETA+DPIT1*COSTHETA+GRAVITY
      XH = XEARTH*COSPSIH + YEARTH*SINPSIH
      YH = -XEARTH*SINPSIH + YEARTH*COSPSIH
      ZH = ZEARTH
```

```
*
*   THE FOLLOWING SECTION CALCULATES THE H-FRAME DERIVATIVE VALUES.
      RUHDOT=RAD*XH
      PSIHDOT=VE*SINLAT/RCOSLAT+YH/UH
      WHD=ZH
      WHDOT=WHD
      LATDOT=UE/RAD
      LNGDOT=VE/RCOSLAT-OMEGA
*OMEGA IS THE EARTHS ANGULAR VELOCITY
*FOR FIVE DEGREES OF FREEDOM DELRDOT MUST BE ZERO
      DELRDOT=-WH
      IF(FIVDEG) DELRDOT=0
*
*   THE H-FRAME DERIVATIVE VALUES ARE INTEGRATED BY A FIRST-ORDER
*   EULER INTEGRATION ROUTINE.
*INTEGRATION IS BYPASSED IN IC AND HOLD MODES
*
      IF(IC.OR.HOLD) GO TO 370
      RUH=RUH+RUHDOT*DELTA
      PSIH=PSIH+PSIHDOT*DELTA
      WH=WH+WHD*DELTA
      LA=LA+LATDOT*DELTA
      LN=LN+LNGDOT*DELTA
      DELR=DELR+DELRDOT*DELTA
*
  370 LAT=LA
      LNG=LN
      RAD=R0+DELR
      RADIUS=RAD
      UH=RUH/RAD
*
*THE FOLLOWING SECTION CALCULATES TRANSLATION OF THE BODY PROJECTED
*ONTO THE EARTHS SURFACE,IN FEET.
*LATO AND LNGO ARE REFERENCE LATITUDE AND LONGITUDE OF THE STARTING
*POINT OF FLIGHT
*PSIR IS THE ANGLE IN RADIANS BETWEEN TRUE NORTH AND TOP OF MAP
*POSITIVE PSIR IS WITH MAP ROTATED EAST,(CLOCKWISE) FROM NORTH
*NOTE THAT THE COSLAT TERM ACCOUNTS FOR CHANGE IN DISTANCE BETWEEN
*LONGITUDE LINES AS A FUNCTION OF LATITUDE
      SINLAT=DSIN(LA)
      COSLAT=DCOS(LA)
      RCOSLAT=RAD*COSLAT
      SINPSIR=SIN(PSIR)
      COSPSIR=COS(PSIR)
      DOWNRNG=R0*((LAT-LATO)*COSPSIR+(LNG-LNGO)*SINPSIR*COSLAT)
      CROSSRNG=R0*((LAT-LATO)*SINPSIR+(LNG-LNGO)*COSPSIR*COSLAT)
      ALTITUDE=DELR+R0*(.00167828-.00167616*COS(2*LAT)-.00000211*
     * COS(4*LAT))
      COSPSIH=DCOS(PSIH)
      SINPSIH=DSIN(PSIH)
      UE=UH*COSPSIH
      UEARTH=UE
      VE=UH*SINPSIH
      VEARTH=VE
      WE=WH
      WEARTH=WE
*
*   WINDS
*   NORTH WINDS(NWIND) ANS EAST WIND (EWIND) ARE TABLE LOOKUPS
```

```
*    AS A FUNCTION OF ALTITUDE AND ARE ADDED TO CORRESPONDING EARTH
*    AXIS VELOCITIES TO GET AIR SPEED
        IF(.NOT.WIND) GO TO 410
        WINDALT = ALTITUDE
        IF (WINDALT.GT. 95000) WINDALT = 95000
        CALL POINTF(WINDALT,20,PLACE,FRACTION,WINDARG)
        NWIND=NWTABL(PLACE)+(NWTABL(PLACE+1)-NWTABL(PLACE))*FRACTION
        EWIND=EWTABL(PLACE)+(EWTABL(PLACE+1)-EWTABL(PLACE))*FRACTION
        GO TO 420
  410 NWIND = EWIND = 0
* EARTH AXIS AND AIR VELOCITIES CONSIDER H-FRAME VELOCITIES,
* WINDS, AND ROTATING EARTH
  420 UA=UE-NWIND
        UAIR=UA
        VA=VE-EWIND-RCOSLAT*OMEGA
        VAIR=VA
        HDOT = -WH
*HDOT REFERENCES POSITIVE UPWARDS
*
* CALCULATION OF BODY-AXIS VELOCITIES, ALPHA, BETA
* SIGN CONVENTIONS FOR VELOCITIES ARE, U IS POSITIVE FOR FORWARD
* V IS POSITIVE RIGHT, AND W IS POSITIVE DOWNWARD
        DPIT1=VA*COSPSI-UA*SINPSI
        DPIT2=UA*COSPSI+VA*SINPSI
        DPIT3=WE*COSTHETA+DPIT2*SINTHETA
        UB=DPIT2*COSTHETA-WE*SINTHETA
        UBODY=UB
        VB=DPIT1*COSPHI+DPIT3*SINPHI
        VBODY=VB
        WB=DPIT3*COSPHI-DPIT1*SINPHI
        WBODY=WB
*    ALPHA IS ANGLE OF ATTACK IN RADIANS
        ALPHA=ATAN2(WBODY,UBODY)
        SINALF=SIN(ALPHA)
        COSALF=COS(ALPHA)
        PIT1 = SQRT (UBODY**2 + WBODY**2)
*    BETA IS ANGLE OF SIDESLIP IN DEGREES
        BETA=ATAN2(VBODY,PIT1)
        BETA=BETA*57.29577951
*VELTRUE IS TRUE AIRSPEED IN FT/SEC
*TEST FOR FIVE DEGREE-OF-FREEDOM
        IF(FIVDEG) GO TO 430
        VELTRUE=SQRT(VBODY**2+PIT1**2)
        GO TO 440
  430 VELTRUE=VELIC
  440 CONTINUE
*
*RHO AND QBAR CALCULATIONS REQUIRE TABLE LOOKUP OF ZETA
        GEOALT=RE*ALTITUDE/(RE+ALTITUDE)
        RHOALT=GEOALT
        CALL POINTF(RHOALT,41,PLACE,FRACTION,ZETARG)
        ZETA=ZETABL(PLACE)+(ZETABL(PLACE+1)-ZETABL(PLACE))*FRACTION
        RHO=RHO0*EXP(-ZETA*GEOALT/10000)
        QBAR = .5*RHO*VELTRUE**2
*EQUIVALENT AND CALIBRATED AIRSPEEDS ARE CALCULATED IN KNOTS
*KEAS IS EQUIVALENT AIRSPEED, KCAS IS CALIBRATED AIRSPEED
        KEAS = VELTRUE*SQRT(RHO/RHO0)*KNOTCON
*THE CALCULATION OF MACH REQUIRES A TABLE LOOKUP OF SPEED OF
*SOUND BASED ON ALTITUDE
```

```
      CALL POINTF(RHOALT,31,PLACE,FRACTION,SNDARG)
      SOUND=SNDTABL(PLACE)+(SNDTABL(PLACE+1)-SNDTABL(PLACE))*FRACTION
      MACH = VELTRUE/SOUND
*REAL FUNCTION SPEED IS USED TO DERIVE KCAS FROM KEAS, AND
*IS PRESENTED IN THE FUNCTION DEFINITION SECTION
      PIT1 = SPEED(MACH)
      PIT2 = SPEED(.001511*KCAS)
      KCAS = PIT1*KEAS/PIT2
*
*
*    ANALOG OUTPUT AND DIGITAL INPUT ARE STARTED HERE.
*
      INLINE
         LW        4,8Z0003FFFF
         LEA       1,)915
         STMW      1,256
         CD        0,428000
      ENDI
*
*
*    IF DYNAMIC CHECKS ARE DESIRED THE APPROPRIATE SURFACES ARE DEFLECTED
*    AND FCS PROCESSING IS BYPASSED.
*
      IF(.NOT.DEPULSE) GO TO 443
      DEL=DELIC+PULSE*16.0
      DER=DERIC+PULSE*16.0
      GO TO 448
  443 IF(.NOT.DAPULSE) GO TO 444
      DEL=DELIC+PULSE*16.0
      DER=DERIC-PULSE*16.0
      GO TO 448
  444 IF(.NOT.DRPULSE) GO TO 445
      DRUD=DRIC+PULSE*20.0
      DER=DERIC
      DEL=DELIC
      GO TO 449
  445 CONTINUE
*
*EXECUTE SHUTTLE FLIGHT CONTROL SYSTEM
*
      CALL SHTLFCS
*
      IF(.NOT.(FCSDE.OR.FCSDA)) GO TO 448
      DEL=DELIC
      DER=DERIC
  448 IF(FCSDR) DRUD=DRIC
  449 IF(FCSDF) FLAP=DBFIC
      IF(FCSSB) BRAK=DSBIC
      ELEV=(DEL+DER)/2
      DAIL=(DEL-DER)/2
*
*    THIS SECTION BUILDS THE DATA HALFWORD TABLES TO BE OUTPUT THRU
*    THE D-A CONVERTER.
      JJ=1
      DATA=QDOT
      INLINE
         BL        )950
      ENDI
      JJ=2
```

```
DATA=PDOT
INLINE
   BL        )950
ENDI
JJ=3
DATA=RDOT
INLINE
   BL        )950
ENDI
JJ=4
DATA=QBODY
INLINE
   BL        )950
ENDI
JJ=5
DATA=PBODY
INLINE
   BL        )950
ENDI
JJ=6
DATA=RBODY
INLINE
   BL        )950
ENDI
JJ=7
DATA=AZ
INLINE
   BL        )950
ENDI
JJ=8
DATA=AY
INLINE
   BL        )950
ENDI
JJ=9
DATA=AX
INLINE
   BL        )950
ENDI
JJ=10
DATA=ALPHA
INLINE
   BL        )950
ENDI
JJ=11
DATA=BETA
INLINE
   BL        )950
ENDI
JJ=12
DATA=QBAR
INLINE
   BL        )950
ENDI
JJ=13
DATA=VELTRUE
INLINE
   BL        )950
ENDI
```

```
JJ=14
DATA=MACH
INLINE
   BL        )950
ENDI
JJ=15
DATA=PHI
INLINE
   BL        )950
ENDI
JJ=16
DATA=PSI
INLINE
   BL        )950
ENDI
JJ=17
DATA=THETA
INLINE
   BL        )950
ENDI
JJ=18
DATA=ALTITUDE
INLINE
   BL        )950
ENDI
JJ=19
DATA=KEAS
INLINE
   BL        )950
ENDI
JJ=20
DATA=VELTRUE
INLINE
   BL        )950
ENDI
JJ=21
DATA=HDOT
INLINE
   BL        )950
ENDI
JJ=22
DATA=WHDOT
INLINE
   BL        )950
ENDI
JJ=23
DATA=CROSSRNG
INLINE
   BL        )950
ENDI
JJ=24
DATA=DOWNRNG
INLINE
   BL        )950
ENDI
JJ=25
DATA=LAT
INLINE
   BL        )950
```

```
      ENDI
      JJ=26
      DATA=LNG
      INLINE
         BL        )950
      ENDI
      JJ=27
      DATA=DER
      INLINE
         BL        )950
      ENDI
      JJ=28
      DATA=ELEV
      INLINE
         BL        )950
      ENDI
      JJ=29
      DATA=DAIL
      INLINE
         BL        )950
      ENDI
      JJ=30
      DATA=ALTITUDE
      INLINE
         BL        )950
      ENDI
      JJ=31
      DATA=DEL
      INLINE
         BL        )950
      ENDI
      JJ=32
      DATA=DRUD
      INLINE
         BL        )950
      ENDI
      JJ=33
      DATA=BRAK
      INLINE
         BL        )950
      ENDI
      JJ=34
      DATA=FLAP
      INLINE
         BL        )950
      ENDI
      JJ=35
      DATA=UZCMD
      INLINE
         BL        )950
      ENDI
*
*
*   'READY' IS USED TO INDICATE THAT A NEW LOOP IS TO START.
*   IT IS SET TO .TRUE. BY THE INTERVAL TIMER INTERRUPT HANDLER.
      READY=.FALSE.
*
*TEST FOR DESIRED EASE
*EASE IS REQUESTED BY SETTING SENSE SWITCH 0 = .TRUE.
```

```
*
  500 CALL SSWTCH(12,EASER)
      IF(.NOT.EASER) GO TO 550
  505 CONTINUE
      INLINE
        DI       43              DISABLE INTERVAL TIMER
      ENDI
*   C ANGLES ARE FOR COMMUNICATIONS TO AND FROM THE OPERATOR VIA EASE.
*   THEY ARE EXPRESSED IN DEGREES.
      CTHETA=THETA*57.29577951
      CPHI=PHI*57.29577951
      CPSI=PSI*57.29577951
      CALPHA=ALPHA*57.29577951
      CPSIH=PSIH*57.29577951
      CPSIR=PSIR*57.29577951
      CMU=MU*57.29577951
      CLAT=LAT*57.29577951
      CLNG=LNG*57.29577951
      CALL M:LOAD ('EASEFCS ',II)
      CALL EASE(DATADBL,DBLSIZE,DATAWRDR,RSIZE,DATAWRDI,ISIZE,
     * DATAHAF,HAFSIZE,DATABYT,BYTSIZE,LIST,SIZE)
      GO TO 10
*
*WAIT FOR INTERVAL TIMER INTERRUPT
  550 TIME=TIME+DELTA
      IF(TIME.GT.RUNTIME) GO TO 505
  551 CONTINUE
      IF(READY) GO TO 47
      GO TO 551
*
*   THIS IS THE ANALOG INPUT IOCB
      INLINE
)910     DATAW    8ZC0010020
         ACH      )911
         ACH      DATAIN
         RES      1W
)911     RES      1W
*
*   THIS IS THE LINKED IOCB FOR BOTH ANALOG INPUT AND DIGITAL OUTPUT
)915     DATAW    8Z4200002C
         ACH      )916
         ACH      DATAOUT
         RES      1W
)920     DATAW    8ZE0000001
         ACH      )921
         ACH      DISCRTIN
         RES      1W
)916     RES      1W
)921     DATAW    8Z40000000
      ENDI
*
      INLINE
)950     STW      0,LINK
      ENDI
      IF(SCALOUT(JJ).LT.0) GO TO 954
      IF(DATA.LE.SCALOUT(JJ)) GO TO 951
      DATAOUT(JJ)=32767
      GO TO 953
  951 IF(DATA.GE.-SCALOUT(JJ)) GO TO 952
```

```
      DATAOUT(JJ)=-32767
      GO TO 953
  952 DATAOUT(JJ)=DATA/SCALOUT(JJ)*32767
      INLINE
)953      LW       3,JJ
          SLA      3,1
          LH       7,DATAOUT-2,3
          SRL      7,4
          STH      7,DATAOUT-2,3
          LW       0,LINK
          TRSW     0
      ENDI
  954 IF(DATA.GE.SCALOUT(JJ)) GO TO 955
      DATAOUT(JJ)=32767
      GO TO 953
  955 IF(DATA.LE.-SCALOUT(JJ)) GO TO 952
      DATAOUT(JJ)=-32767
      GO TO 953
*
      INLINE
)960      STW      0,LINK
          LW       3,JJ
          SLA      3,1
          LH       7,DATAIN-2,3
          SLL      7,1
          STH      7,DATAIN-2,3
      ENDI
      DATA=DATAIN(JJ)*SCALIN(JJ)/32767
      INLINE
          LW       0,LINK
          TRSW     0
      ENDI
*
*
      INLINE
)1000     ACW      )1010
)1010     RES      1W
          STW      7,SAVFL
          LB       7,2ZFF
          STB      7,READY
          LW       7,SAVFL
          BRI      *)1010
      ENDI
*
*
      INLINE
)1100     DI       43             DISABLE INTERVAL TIMER
          LW       7,HANDLER
          STW      7,428
      ENDI
      STOP
*
      INLINE
)1105     DI       43                   DISABLE INTERVAL TIMER
          LEA      6,)1105
          LW       7,8ZFFFFFFEC
          CALM     84
      ENDI
      GO TO 10
      END
```

APPENDIX B - EASE PROGRAM LISTING

```
*                                                                         *
*                              E A S E                                    *
*                                                                         *
***************************************************************************
      SUBROUTINE EASE(DATADBL,DBLSIZE,DATAWRDR,RSIZE,DATAWRDI,ISIZE,
     1 DATAHAF,HAFSIZE,DATABYT,BYTSIZE,LIST,SIZE)
*     THE PURPOSE OF THE EASE PROGRAM IS TO ALLOW INPUT AND OUTPUT OF
*     DATA VALUES. DATA MAY BE REFERENCED BY EITHER VARIABLE NAME OR
*     NUMBERED MEMORY LOCATION.
*
*     DATA IS COMMUNICATED IN THE FOLLOWING FASHION-
*
*
*     NAMED VARIABLES ARE ADDRESSED BY NAME, AND MEMORY LOCATIONS ARE
*     ADDRESSED BY A FULLWORD ADDRESS.
*
*     TO EXAMINE A DATA VALUE A QUESTION MARK SHOULD FOLLOW THE LOCATION
*     DESCRIPTION.
*
*     TO CHANGE THE VALUE OF THE SPECIFIED LOCATION AN EQUAL SIGN SHOULD
*     FOLLOW THE LOCATION DESCRIPTION.
*
*     INQUIRIES OF OR INPUTS TO MEMORY LOCATIONS THAT ARE INTEGER VALUES
*     SHOULD BE PRECEDED BY AN ASTERISK.
*     INQUIRIES OF OR INPUTS TO MEMORY LOCATIONS THAT ARE REAL VALUES MAY
*     BE LEFT UNPREFIXED.
*
*     IMBEDDED BLANKS ARE IGNORED IN ALL CASES.
*
*     TO RETURN TO THE MAIN PROGRAM AN EXCLAMATION MARK SHOULD BE TYPED.
*
*     HEX ADDRESSES BEGINNING WITH A LETTER (C.F. A72C) MUS BE PRECEEDED
*     BY A ZERO (C.F. 0A72C)
*
*
*     THE FOLLOWING INPUT WILL CAUSE ALFA TO BE SET TO 30 DEGREES-
*     ALFA=30
*     THE FOLLOWING WILL CAUSE THE PROGRAM TO OUTPUT ZETA-
*     ZETA?
*     THE FOLLOWING WILL CAUSE THE PROGRAM TO OUTPUT ARE REAL VALUE FROM
*     LOCATION 12345
*     12345?
*     THE FOLLOWING WILL INPUT THE INTEGER VALUE OF 546 TO LOCATION 12345
*     *12345=456
*
*
*     THE DATA STORAGE ARRAYS ARE SET UP AS FOLLOWS
*
*****************************************************************    K=1
*                                                               *
*                                                               *
*          DATADBL(K)=REAL DOUBLEWORD ARRAY                     *
```

```
*                                                                  *
*                                                                  *
********************************************************************  K=KMAXDBL
*                                                                  *
*                                                                  *
*         DATAWRDR(K)=REAL WORD ARRAY                              *
*                                                                  *
*                                                                  *
********************************************************************  K=KMAXREAL
*                                                                  *
*                                                                  *
*         DATAWRDI(K)=INTEGER WORD ARRAY                           *
*                                                                  *
*                                                                  *
********************************************************************  K=KMAXWRD
*                                                                  *
*                                                                  *
*         DATAHAF(K)=INTEGER HALFWORD ARRAY                        *
*                                                                  *
*                                                                  *
********************************************************************  K=KMAXHAF
*                                                                  *
*                                                                  *
*         DATABYT(K)=INTEGER BYTE ARRAY                            *
*                                                                  *
*                                                                  *
********************************************************************  K=SIZE
*
*
*    THE LIST SYMBOL ARRAY CONTAINS A DOUBLEWORD CHARACTER NAME FOR EACH
*    VARIABLE IN THE DATA(K) ARRAY.  THE VALUE OF K IN LIST(K) CORRESPOND
*    TO THE VALUE OF K IN THE DATA(K) ARRAYS
*
*
*    THE FOLLOWING VARIABLES MUST BE DIMENSIONED AS INDICATED TO INSURE
*    PROPER OPERATION OF THE PROGRAM IN EACH CASE
*    DATABYT(VALUE OF KMAXHAF,SIZE)
*    DATAHAF(VALUE OF KMAXWRD,VALUE OF KMAXHAF)
*    DATAWRDI(VALUE OF KMAXREAL,VALUE OF KMAXWRD)
*    DATAWRDR(VALUE OF KMAXDBL,VALUE OF KMAXREAL)
*    DATADBL(1,VALUE OF KMAXDBL)
*    LIST(TOTAL NUMBER OF VARIABLES)
*
*
*    ALSO, THE FOLLOWING PARAMETERS MUST BE INITIALIZED IN A DATA
*    STATEMENT
*    SIZE=TOTAL NUMBER OF VARIABLES
*    KMAXDBL=VALUE OF K AT END OF DOUBLEWORD ARRAY
*    KMAXREAL=VALUE OF K AT END OF REAL WORD ARRAY
*    KMAXWRD=VALUE OF K AT END OF INTEGER WORD ARRAY
*    KMAXHAF=VALUE OF K AT END OF HALFWORD ARRAY
*
*
      BIT INTGR,LETTER,DECIMAL
      CHAR WORD(81),TEMP(8),NUMB(5),OUTBUF(25),NAMEIN(25)
      CHAR MSSG1(15),MSSG2(18),MSSG3(34),MSSG4(33),MSSG5(35),MSSG6(31)
      CHAR MSSG7(37),MSSG8(36),MSSG9(29),MSSG10(22)
      INTEGER*2 DBLSIZE,RSIZE,ISIZE,HAFSIZE,BYTSIZE,SIZE
      INTEGER*1 DATABYT(BYTSIZE)
```

```
      INTEGER*2 DATAHAF(HAFSIZE)
      INTEGER*4 DATAWRDI(ISIZE)
      INTEGER*4 ADDR,BOXI
      INTEGER*8 LIST(SIZE),INBUFI,SYMBOL
      REAL*4 DATAWRDR(RSIZE),BOXR
      REAL*8 DATADBL(DBLSIZE),INBUFR
      EQUIVALENCE (SYMBOL,TEMP(1)),(TEMP(1),NUMB(1)),(BOXR,BOXI)
      DATA MSSG1/'WELCOME TO EASE'/
      DATA MSSG2/'WRONG INPUT FORMAT'/
      DATA MSSG3/'VARIABLE NAME EXCEEDS 8 CHARACTERS'/
      DATA MSSG4/'NO SUCH SYMBOL IS DEFINED TO EASE'/
      DATA MSSG5/'MEMORY ADDRESS MUST BE A HEX NUMBER'/
      DATA MSSG6/'MEMORY ADDRESS EXCEEDS 5 DIGITS'/
      DATA MSSG7/'MEMORY ADDRESS IS OUTSIDE CORE LIMITS'/
      DATA MSSG8/'MEMORY DATA MUST BE A DECIMAL NUMBER'/
      DATA MSSG9/'MEMORY DATA EXCEEDS 25 DIGITS'/
      DATA MSSG10/'END OF EASE PROCESSING'/
      KMAXDBL=DBLSIZE
      KMAXREAL=DBLSIZE+RSIZE
      KMAXWRD=DBLSIZE+RSIZE+ISIZE
      KMAXHAF=DBLSIZE+RSIZE+ISIZE+HAFSIZE
      CALL CARRIAGE
      CALL M:TELEW(MSSG1,15)
      CALL CARRIAGE
   10 CONTINUE
      LETTER=INTGR=.FALSE.
      CALL CARRIAGE
      CALL M:TELER(WORD,80)
      L=2
*    TEST FOR BLANKS IN INPUT LINE
   15 IF(WORD(L).NE.' ') GO TO 20
      L=L+1
      GO TO 15
   20 IF(WORD(L).EQ.'!' ) GO TO 100
      IF((WORD(L).LT.'A').OR.(WORD(L).GT.'Z')) GO TO 22
      GO TO 200
   22 IF(WORD(L).NE.'*') GO TO 24
      INTGR=.TRUE.
   23 L=L+1
      IF(WORD(L).NE.' ') GO TO 24
      GO TO 23
   24 IF((WORD(L).LT.'0').OR.(WORD(L).GT.'9')) GO TO 30
      GO TO 300
   40 CONTINUE
      IF(WORD(L).EQ.'?' ) GO TO 400
      IF(WORD(L).EQ.'=') GO TO 500
      GO TO 30
*
*
*    THE FOLLOWING SECTION CONTAINS ERROR MESSAGES
*
   30 CALL CARRIAGE
      CALL M:TELEW(MSSG2,18)
      GO TO 10
*
   50 CALL CARRIAGE
      CALL M:TELEW(MSSG3,34)
      GO TO 10
*
```

```
   60 CALL CARRIAGE
      CALL M:TELEW(MSSG4,33)
      GO TO 10
*
   70 CALL CARRIAGE
      CALL M:TELEW(MSSG5,35)
      GO TO 10
*
   80 CALL CARRIAGE
      CALL M:TELEW(MSSG6,31)
      GO TO 10
*
   90 CALL CARRIAGE
      CALL M:TELEW(MSSG7,37)
      GO TO 10
*
   55 CALL CARRIAGE
      CALL M:TELEW(MSSG8,36)
      GO TO 10
*
   65 CALL CARRIAGE
      CALL M:TELEW(MSSG9,29)
      GO TO 10
*
*
*
*    NAMED VARIABLE PROCESSING
  200 CONTINUE
*    LETTER SIGNIFIES TO THE I/O SECTION THAT THE EFFECTIVE VARIABLE
*    ADDRESS IS DEFINED BY LOCATION IN THE VARIABLE TABLE.
      LETTER=.TRUE.
*    INITIALIZE TEMP TO BLANKS
      TEMP=' '
      J=1
  210 TEMP(J)=WORD(L)
  220 L=L+1
      IF(WORD(L).NE.' ') GO TO 230
      GO TO 220
  230 IF((WORD(L).EQ.'?' ).OR.(WORD(L).EQ.'=')) GO TO 250
      J=J+1
      IF(J.EQ.9) GO TO 50
      GO TO 210
*    TEST FOR SYMBOL IN SYMBOL LIST AND FIND RELATIVE LOCATION IN DATA
*    ARRAYS
  250 K=1
  255 IF(SYMBOL.EQ.LIST(K)) GO TO 260
      K=K+1
      IF(K.GT.SIZE) GO TO 60
      GO TO 255
  260 IF(K.GT.KMAXREAL) INTGR=.TRUE.
*    RETURN TO MAIN
      GO TO 40
*
*
*
*    NUMBERED ADDRESS PROCESSING
  300 CONTINUE
*    FILL NUMB WITH BLANKS
      NUMB=' '
```

```
      J=1
  310 NUMB(J)=WORD(L)
  320 L=L+1
      IF(WORD(L).NE.' ') GO TO 330
      GO TO 320
  330 IF((WORD(L).EQ.'?' ).OR.(WORD(L).EQ.'=')) GO TO 350
      IF((WORD(L).GE.'0').AND.(WORD(L).LE.'9')) GO TO 340
      IF((WORD(L).GE.'A').AND.(WORD(L).LE.'F')) GO TO 340
      GO TO 70
  340 J=J+1
      IF(J.EQ.6) GO TO 80
      GO TO 310
*    NUMB MUST BE RIGHT JUSTIFIED FOR CONVERSION
  350 IF(J.EQ.5) GO TO 357
      MOVE=5-J
      DO 355 N=1,MOVE
      NUMB(5)=NUMB(4)
      NUMB(4)=NUMB(3)
      NUMB(3)=NUMB(2)
      NUMB(2)=NUMB(1)
  355 NUMB(1)=' '
*    CONVERT ASCII INPUT TO HEX ADDRESS
  357 DECODE(5,360,NUMB)ADDR
  360 FORMAT(Z5)
*    TEST FOR EXISTENT MEMORY ADDRESS
      IF(ADDR.GT.5Z27FFC) GO TO 90
*    RETURN TO MAIN
      GO TO 40
*
*
*
*    CRT OUTPUT PROCESSING
*    TEST TO DETERMINE IF NUMBERED LOCATION OR NAMED VARIABLE OUTPUT IS
*    DESIRED
  400 CONTINUE
      IF(LETTER) GO TO 450
*    RETIEVE VALUE FROM EFFECTIVE ADDRESS
      INLINE
         LW      7,*ADDR
         STW     7,BOXR
      ENDI
*    TEST FOR INTEGER OR REAL CONVERSION
      IF(INTGR) GO TO 410
      ENCODE(25,405,OUTBUF)BOXR
  405 FORMAT(F25.10)
      GO TO 490
  410 ENCODE(25,415,OUTBUF)BOXI
  415 FORMAT(I25)
      GO TO 490
  450 IF(INTGR) GO TO 470
*    TEST TO DETERMINE WHICH ARRAY TYPE IS TO BE ACCESSED, AND PROCESS
*    ACCORDINGLY
      IF(K.GT.KMAXDBL) GO TO 460
      ENCODE(25,405,OUTBUF)DATADBL(K)
      GO TO 490
  460 ENCODE(25,405,OUTBUF)DATAWRDR(K-KMAXDBL)
      GO TO 490
  470 IF(K.GT.KMAXHAF) GO TO 485
      IF(K.GT.KMAXWRD) GO TO 480
```

```
      ENCODE(25,415,OUTBUF)DATAWRDI(K-KMAXREAL)
      GO TO 490
  480 ENCODE(25,415,OUTBUF)DATAHAF(K-KMAXWRD)
      GO TO 490
  485 ENCODE(25,415,OUTBUF)DATABYT(K-KMAXHAF)
*    OUTPUT ANSWER
  490 CALL CARRIAGE
      CALL M:TELEW(OUTBUF,25)
*    RETURN TO MAIN
      GO TO 10
*
*
*
*    CRT INPUT PROCESSING
*    FINISH INPUT OF DESIRED VALUE AND STORE
  500 CONTINUE
      DECIMAL=.FALSE.
      J=1
      NAMEIN=' '
  510 L=L+1
*    TEST FOR BLANKS
      IF(WORD(L).EQ.' ') GO TO 510
*    TEST FOR CARRIAGE RETURN
      IF(WORD(L).EQ.2Z0D) GO TO 520
*    TEST FOR VALID NUMBER
      IF((WORD(L).GE.'0').AND.(WORD(L).LE.'9')) GO TO 515
      IF(WORD(L).EQ.'-') GO TO 515
      IF(WORD(L).NE.'.') GO TO 55
      DECIMAL=.TRUE.
  515 NAMEIN(J)=WORD(L)
      J=J+1
      IF(J.EQ.26) GO TO 65
      GO TO 510
*    NAMEIN NOW CONTAINS THE VALUE TO BE STORED IN ASCII
*    TEST FOR INTEGER OR REAL CONVERSION
  520 IF(INTGR) GO TO 525
      IF(.NOT.DECIMAL) NAMEIN(J)='.'
      DECODE(25,405,NAMEIN)INBUFR
      GO TO 530
  525 IF(J.EQ.25) GO TO 529
*    NAMEIN MUST BE RIGHT JUSTIFIED FOR CONVERSION
      MOVE=26-J
      DO 527 I=1,MOVE
      DO 526 N=1,24
  526 NAMEIN(26-N)=NAMEIN(25-N)
  527 NAMEIN(1)=' '
  529 DECODE(25,415,NAMEIN)INBUFI
*    INBUFR CONTAINS THE REAL DATA VALUE, AND INBUFI CONTAINS THE INTEGER
*    DATA VALUE
*    TEST WHETHER DATA IS FOR NAMED VARIABLE OR NUMBERED MEMORY LOCATION
  530 IF(LETTER) GO TO 550
      IF(INTGR) GO TO 540
*    STORE REAL VALUE IN EFFECTIVE ADDRESS
      INLINE
         LW      7,INBUFR
         STW     7,*ADDR
      END
      GO TO 599
*    STORE INTEGER VALUE IN EFFECTIVE ADDRESS
```

```
  540 CONTINUE
      INLINE
         LW      7,INBUFI+1W
         STW     7,*ADDR
      ENDI
      GO TO 599
*   CHOOSE DATA TABLE IN WHICH VARIABLE IS TO BE STORED
  550 IF(K.GT.KMAXHAF) GO TO 560
      IF(K.GT.KMAXWRD) GO TO 562
      IF(K.GT.KMAXREAL) GO TO 564
      IF(K.GT.KMAXDBL) GO TO 566
      DATADBL(K)=INBUFR
      GO TO 570
  560 DATABYT(K-KMAXHAF)=INBUFI
      GO TO 570
  562 DATAHAF(K-KMAXWRD)=INBUFI
      GO TO 570
  564 DATAWRDI(K-KMAXREAL)=INBUFI
      GO TO 570
  566 DATAWRDR(K-KMAXDBL)=INBUFR
*   THE FOLLOWING SECTION TEST FOR VALUES OF K THAT REQUIRE ADDITIONAL
*   CALCULATIONS FOR EACH CHANGE IN VALUE
  570 CONTINUE
*
*
*        INSERT ADDED CALCULATIONS HERE    *******************
***************
****************
*   RETURN TO MAIN
  599 GO TO 10
*
*
  100 CALL CARRIAGE
      CALL M:TELEW(MSSG10,22)
      RETURN
*
*   END OF EASE PROGRAM
      END
```

```
      PROGRAM DATASTOR
      REAL DATA(1000)
      INTEGER*8 NAME,NAMELIST(45)
*
*    NOTE THAT OUTDATA MUST BE DIMENSIONED TO 'SIZE', OR THE TOTAL
*    NUMBER OF WORDS IN ALL DATA ARRAYS.
      REAL*4 OUTDATA(5243)
*
      INTEGER*4 LENGTH,NUMBER,DIFF,ADDRESS,SIZE,PIT,STAT
*
      REAL*4      CCOT(63)
      REAL*4      DCCET(378)
      REAL*4 CCDBF1T(63)
      REAL*4      CCDBF2T(63)
      REAL*4      CCDSB1T(63)
      REAL*4      CCDSB2T(63)
      REAL*4      CLBTABL(252)
      REAL*4      CLBDE1T(63)
      REAL*4      CLBDE2T(63)
      REAL*4      CLBDSB1T(63)
      REAL*4      CLBDSB2T(63)
      REAL*4      CLBDSB3T(63)
      REAL*4      CLDAT(378)
      REAL*4      CLDRT(252)
      REAL*4      CLPT(63)
      REAL*4      CLRT(63)
      REAL*4      CMOT(63)
      REAL*4      DCMET(378)
      REAL*4      CMDBF1T(63)
      REAL*4      CMDBF2T(63)
      REAL*4      CMDSB1T(63)
      REAL*4      CMDSB2T(63)
      REAL*4      CMQT(252)
      REAL*4      CNBTABL(252)
      REAL*4      CNBDE1T(63)
      REAL*4      CNBDE2T(63)
      REAL*4      CNBDSB1T(63)
      REAL*4      CNBDSB2T(63)
      REAL*4      CNBDSB3T(63)
      REAL*4      CNDAT(378)
      REAL*4      CNDRT(252)
      REAL*4      CNDRBT(63)
      REAL*4      CNPT(63)
      REAL*4      CNRT(63)
      REAL*4      CNOT(63)
      REAL*4      CNDET(63)
      REAL*4      CNDBF1T(63)
      REAL*4      CNDBF2T(63)
      REAL*4      CNDSB1T(63)
      REAL*4      CNDSB2T(63)
      REAL*4      CYBT(63)
      REAL*4      CYDAT(378)
      REAL*4      CYDRT(63)
      REAL*4      CYDRSB1T(7)
      REAL*4      CYDRSB2T(7)
```

```
      CHAR BUFFER(81),COMMENTS(65),MSSG(80), TEST
      CHAR MSSG1(45),MSSG2(44),MSSG3(64)
      EQUIVALENCE (BUFFER(2), MSSG(1), TEST)
      EQUIVALENCE
     * (OUTDATA(1),CCOT(1))
     *,(OUTDATA(64),DCCET(1))
     *,(OUTDATA(442),CCDBF1T(1))
     *,(OUTDATA(505),CCDBF2T(1))
     *,(OUTDATA(568),CCDSB1T(1))
     *,(OUTDATA(631),CCDSB2T(1))
     *,(OUTDATA(694),CLBTABL(1))
     *,(OUTDATA(946),CLBDE1T(1))
     *,(OUTDATA(1009),CLBDE2T(1))
     *,(OUTDATA(1072),CLBDSB1T(1))
     *,(OUTDATA(1135),CLBDSB2T(1))
     *,(OUTDATA(1198),CLBDSB3T(1))
     *,(OUTDATA(1261),CLDAT(1))
     *,(OUTDATA(1639),CLDRT(1))
     *,(OUTDATA(1891),CLPT (1))
     *,(OUTDATA(1954),CLRT (1))
     *,(OUTDATA(2017),CMOT (1))
     *,(OUTDATA(2080),DCMET(1))
     *,(OUTDATA(2458),CMDBF1T(1))
     *,(OUTDATA(2521),CMDBF2T(1))
     *,(OUTDATA(2584),CMDSB1T(1))
     *,(OUTDATA(2647),CMDSB2T(1))
     *,(OUTDATA(2710),CMQT(1))
     *,(OUTDATA(2962),CNBTABL(1))
     *,(OUTDATA(3214),CNBDE1T(1))
     *,(OUTDATA(3277),CNBDE2T(1))
     *,(OUTDATA(3340),CNBDSB1T(1))
     *,(OUTDATA(3403),CNBDSB2T(1))
     *,(OUTDATA(3466),CNBDSB3T(1))
     *,(OUTDATA(3529),CNDAT(1))
     *,(OUTDATA(3907),CNDRT(1))
     *,(OUTDATA(4159),CNDRBT(1))
     *,(OUTDATA(4222),CNPT(1))
     *,(OUTDATA(4285),CNRT(1))
     *,(OUTDATA(4348),CNOT(1))
     *,(OUTDATA(4411),CNDET(1))
     *,(OUTDATA(4474),CNDBF1T(1))
     *,(OUTDATA(4537),CNDBF2T(1))
     *,(OUTDATA(4600),CNDSB1T(1))
     *,(OUTDATA(4663),CNDSB2T(1))
     *,(OUTDATA(4726),CYBT(1))
     *,(OUTDATA(4789),CYDAT(1))
     *,(OUTDATA(5167),CYDRT(1))
     *,(OUTDATA(5230),CYDRSB1T(1))
     *,(OUTDATA(5237),CYDRSB2T(1))
```

```
      DATA MSSG1/'IF HEADER CARD CORRECT TYPE C, IF NOT TYPE A?'/
      DATA MSSG2 /'PLEASE INPUT CORRECT HEADER CARD INFORMATION'/
      DATA MSSG3/'NAME WAS NOT FOUND IN NAMELIST PLEASE ENTER CORRECT IN
     1FORMATION.'/
      DATA SIZE/5243/
      DATA NUMBER/45/
*
      DATA NAMELIST/
     *  'CCO'
     * ,'CCE'
     * ,'CCDBF1'
     * ,'CCDBF2'
     * ,'CCDSB1'
     * ,'CCDSB2'
     * ,'CLB'
     * ,'CLBDE1'
     * ,'CLBDE2'
     * ,'CLBDSB1'
     * ,'CLBDSB2'
     * ,'CLBDSB3'
     * ,'CLDA'
     * ,'CLDR'
     * ,'CLP'
     * ,'CLR'
     * ,'CMO'
     * ,'CME'
     * ,'CMDBF1'
     * ,'CMDBF2'
     * ,'CMDSB1'
     * ,'CMDSB2'
     * ,'CMQ'
     * ,'CNB'
     * ,'CNBDE1'
     * ,'CNBDE2'
     * ,'CNBDSB1'
     * ,'CNBDSB2'
     * ,'CNBDSB3'
     * ,'CNDA'
     * ,'CNDR'
     * ,'CNDRB'
     * ,'CNP'
     * ,'CNR'
     * ,'CNO'
     * ,'CNDE'
     * ,'CNDBF1'
     * ,'CNDBF2'
     * ,'CNDSB1'
     * ,'CNDSB2'
     * ,'CYB'
     * ,'CYDA'
     * ,'CYDR'
     * ,'CYDRDSB1'
     * ,'CYDRDSB2'/
   10 DATA = 0.0
      READ (4,20,END=150) NAME, LENGTH, COMMENTS
   20 FORMAT (A8,1X,I6,65A1)
   30 ENCODE (80,20,MSSG) NAME, LENGTH,COMMENTS
      IF (TEST.EQ.'C') GO TO 50
```

```
*  THIS SECTION OUTPUTS THE HEADER CARD TO THE CRT
*
      CALL CARRIAGE
      CALL M:TELEW (MSSG,80)
      CALL CARRIAGE
      CALL M:TELEW (MSSG1,45)
      CALL M:TELER(BUFFER,80)
      IF (TEST.EQ.'I') GO TO 150
      IF (TEST.EQ.'C') GO TO 50
      CALL CARRIAGE
      CALL M:TELEW (MSSG2,44)
   40 CALL CARRIAGE
      CALL M:TELEW (MSSG, 80)
      CALL M:TELEW (MSSG, 80)
      CALL CARRIAGE
      CALL M:TELER (BUFFER,81)
      DECODE(80,20,MSSG) NAME, LENGTH, COMMENTS
      GO TO 30
*
*  SEARCH EASE TABLE OF DERIVATIVE NAMES FOR NAME AND RETURN POSITION
*
   50 J=0
   51 J=J+1
      IF (NAME.EQ.NAMELIST(J)) GO TO 55
      IF (J.LT.NUMBER) GO TO 51
      CALL CARRIAGE
      CALL M:TELEW (MSSG3,64)
      GO TO 40
   55 CONTINUE
      INLINE
         LW       3,J
         SLA      3,2
         LW       7,)200-4,3
         STW      7,ADDRESS
      ENDI
*
*THIS SECTION READS THE DATA CARDS UNTIL THE NUMBER OF POINTS READ=LENGT
*
   59 READ (4,60) ( DATA(I),I=1,LENGTH)
   60 FORMAT (8F10.6)
*
      DO 160 I=1,LENGTH
      DATAW = DATA(I)
      INLINE
         LW       3,I
         SLA      3,2
         ADMW     3,ADDRESS
         LW       7,DATAW
         STW      7,-4,3
      ENDI
  160 CONTINUE
      GO TO10
  150 CALL CARRIAGE
```

```
      I=1
      PIT=1152
  180 CALL BUFFEROUT(5,1,OUTDATA(I),PIT)
  185 CALL STATUS(5,STAT)
      IF(STAT.NE.2) GO TO 185
      I=I+1152
      IF(I.GT.SIZE) GO TO 186
      IF(I+1152.GT.SIZE) PIT=SIZE+1-I
      GO TO 180
  186 CONTINUE
      STOP
      INLINE
)200      ACW      CCOT
          ACW      DCCET
          ACW      CCDBF1T
          ACW      CCDBF2T
          ACW      CCDSB1T
          ACW      CCDSB2T
          ACW      CLBTABL
          ACW      CLBDE1T
          ACW      CLBDE2T
          ACW      CLBDSB1T
          ACW      CLBDSB2T
          ACW      CLBDSB3T
          ACW      CLDAT
          ACW      CLDRT
          ACW      CLPT
          ACW      CLRT
          ACW      CMOT
          ACW      DCMET
          ACW      CMDBF1T
          ACW      CMDBF2T
          ACW      CMDSB1T
          ACW      CMDSB2T
          ACW      CMQT
          ACW      CNBTABL
          ACW      CNBDE1T
          ACW      CNBDE2T
          ACW      CNBDSB1T
          ACW      CNBDSB2T
          ACW      CNBDSB3T
          ACW      CNDAT
          ACW      CNDRT
          ACW      CNDRBT
          ACW      CNPT
          ACW      CNRT
          ACW      CNOT
          ACW      CNDET
          ACW      CNDBF1T
          ACW      CNDBF2T
          ACW      CNDSB1T
          ACW      CNDSB2T
          ACW      CYBT
          ACW      CYDAT
          ACW      CYDRT
          ACW      CYDRSB1T
          ACW      CYDRSB2T
      ENDI
      END
```

```
************************************************************************
*                                                                      *
*                D E R I V E     F U N C T I O N S                     *
*                                                                      *
************************************************************************
      FUNCTION DERIVE1(ARRAY,I,ARGPNT1,ARFRAC1)
      INTEGER*2 ARGPNT1,I
      DIMENSION ARRAY(I)
      DERIVE1=(ARRAY(ARGPNT1+1)-ARRAY(ARGPNT1))*ARFRAC1+ARRAY(ARGPNT1)
      RETURN
      END
*
********
*
      FUNCTION DERIVE2(ARRAY2,I,J,ARGPNT1,ARGPNT2,ARFRAC1,ARFRAC2)
      DIMENSION ARRAY2(I,J)
      INTEGER*2 ARGPNT1,ARGPNT2
      TEMP3=(ARRAY2(ARGPNT1+1,ARGPNT2+1)-ARRAY2(ARGPNT1,ARGPNT2+1))*ARFR
     *AC1+ARRAY2(ARGPNT1,ARGPNT2+1)
      TEMP2=(ARRAY2(ARGPNT1+1,ARGPNT2)-ARRAY2(ARGPNT1,ARGPNT2))*ARFRAC1+
     *ARRAY2(ARGPNT1,ARGPNT2)
      DERIVE2=(TEMP3-TEMP2)*ARFRAC2+TEMP2
      RETURN
      END
*
********
*
      FUNCTION DERIVE3(ARRAY3,I,J,K,ARGPNT1,ARGPNT2,ARGPNT3,ARFRAC1,
     *ARFRAC2,ARFRAC3)
      DIMENSION ARRAY3(I,J,K)
      INTEGER*2 ARGPNT1,ARGPNT2,ARGPNT3
      TEMP7=(ARRAY3(ARGPNT1+1,ARGPNT2+1,ARGPNT3+1)-ARRAY3(ARGPNT1,ARGPN
     *T2+1,ARGPNT3+1))*ARFRAC1+ARRAY3(ARGPNT1,ARGPNT2+1,ARGPNT3+1)
      TEMP6=(ARRAY3(ARGPNT1+1,ARGPNT2,ARGPNT3+1)-ARRAY3(ARGPNT1,ARGPNT2,
     *ARGPNT3+1))*ARFRAC1+ARRAY3(ARGPNT1,ARGPNT2,ARGPNT3+1)
      TEMP5=(ARRAY3(ARGPNT1+1,ARGPNT2+1,ARGPNT3)-ARRAY3(ARGPNT1,ARGPNT2+
     *1,ARGPNT3))*ARFRAC1+ARRAY3(ARGPNT1,ARGPNT2+1,ARGPNT3)
      TEMP4=(ARRAY3(ARGPNT1+1,ARGPNT2,ARGPNT3)-ARRAY3(ARGPNT1,ARGPNT2,AR
     *GPNT3))*ARFRAC1+ARRAY3(ARGPNT1,ARGPNT2,ARGPNT3)
      TEMP3=(TEMP7-TEMP6)*ARFRAC2+TEMP6
      TEMP2=(TEMP5-TEMP4)*ARFRAC2+TEMP4
      DERIVE3=(TEMP3-TEMP2)*ARFRAC3+TEMP2
      RETURN
      END
*
********
*
      FUNCTION DERIVE4(ARRAY4,I,J,K,L,ARGPNT1,ARGPNT2,ARGPNT3,ARGPNT4,
     *ARFRAC1,ARFRAC2,ARFRAC3,ARFRAC4)
      DIMENSION ARRAY4(I,J,K,L)
      INTEGER*2 ARGPNT1,ARGPNT2,ARGPNT3,ARGPNT4
*    DERIVE4  IS THE VALUE RERURNED TO THE MAIN PROGRAM.
*    ARRAY4 IS A FOUR DIMENSIOAL DUMMY ARRAY  CONTAINING FUNCTION DATA U
*  I IS AN INTEGER VALUE REPRESENTING THE SIZE OF THE FIRST DIMENSION OF
*  J IS AN INTEGER VALUE REPRESENTING THE SIZE OF THE SECON DIMENSION OF
*  K IS AN INTEGER VALUE REPRESENTING THE SIZE OF THE THIRD DIMENSION OF
*  L IS AN INTEGER VALUE REPRESENTIING FTH SIZE OF THE FOURTH DIMENSION O
```

```
      TEMP15=(ARRAY4(ARGPNT1+1,ARGPNT2+1,ARGPNT3+1,ARGPNT4+1)
     *       -ARRAY4(ARGPNT1,ARGPNT2+1,ARGPNT3+1,ARGPNT4+1))*ARFRAC1
     *       +ARRAY4(ARGPNT1,ARGPNT2+1,ARGPNT3+1,ARGPNT4+1)
      TEMP14=(ARRAY4(ARGPNT1+1,ARGPNT2,ARGPNT3+1,ARGPNT4+1)
     *       -ARRAY4(ARGPNT1,ARGPNT2,ARGPNT3+1,ARGPNT4+1))*ARFRAC1
     *       +ARRAY4(ARGPNT1,ARGPNT2,ARGPNT3+1,ARGPNT4+1)
      TEMP13=(ARRAY4(ARGPNT1+1,ARGPNT2+1,ARGPNT3,ARGPNT4+1)
     *       -ARRAY4(ARGPNT1,ARGPNT2+1,ARGPNT3,ARGPNT4+1))*ARFRAC1
     *       +ARRAY4(ARGPNT1,ARGPNT2+1,ARGPNT3,ARGPNT4+1)
      TEMP12=(ARRAY4(ARGPNT1+1,ARGPNT2,ARGPNT3,ARGPNT4+1)
     *       -ARRAY4(ARGPNT1,ARGPNT2,ARGPNT3,ARGPNT4+1))*ARFRAC1
     *       +ARRAY4(ARGPNT1,ARGPNT2,ARGPNT3,ARGPNT4+1)
      TEMP11=(ARRAY4(ARGPNT1+1,ARGPNT2,ARGPNT3+1,ARGPNT4)
     *       -ARRAY4(ARGPNT1,ARGPNT2+1,ARGPNT3+1,ARGPNT4))*ARFRAC1
     *       +ARRAY4(ARGPNT1,ARGPNT2+1,ARGPNT3+1,ARGPNT4)
      TEMP10=(ARRAY4(ARGPNT1+1,ARGPNT2,ARGPNT3+1,ARGPNT4)
     *       -ARRAY4(ARGPNT1,ARGPNT2,ARGPNT3+1,ARGPNT4))*ARFRAC1
     *       +ARRAY4(ARGPNT1,ARGPNT2,ARGPNT3+1,ARGPNT4)
      TEMP9=(ARRAY4(ARGPNT1+1,ARGPNT2+1,ARGPNT3,ARGPNT4)
     *       -ARRAY4(ARGPNT1,ARGPNT2+1,ARGPNT3,ARGPNT4))*ARFRAC1
     *       +ARRAY4(ARGPNT1,ARGPNT2+1,ARGPNT3,ARGPNT4)
      TEMP8=(ARRAY4(ARGPNT1+1,ARGPNT2,ARGPNT3,ARGPNT4)
     *       -ARRAY4(ARGPNT1,ARGPNT2,ARGPNT3,ARGPNT4))*ARFRAC1
     *       +ARRAY4(ARGPNT1,ARGPNT2,ARGPNT3,ARGPNT4)
      TEMP7=(TEMP15-TEMP14)*ARFRAC2+TEMP14
      TEMP6=(TEMP13-TEMP12)*ARFRAC2+TEMP12
      TEMP5=(TEMP11-TEMP10)*ARFRAC2+TEMP10
      TEMP4=(TEMP9-TEMP8)*ARFRAC2+TEMP8
      TEMP3=(TEMP7-TEMP6)*ARFRAC3+TEMP6
      TEMP2=(TEMP5-TEMP4)*ARFRAC3+TEMP4
      DERIVE4=(TEMP3-TEMP2)*ARFRAC4+TEMP2
      RETURN
      END
************************************************************************
*                                                                      *
*          S U B P R O G R A M     D E F I N I T I O N S               *
*                                                                      *
************************************************************************
*THE FOLLOWING ARE THE FUNCTION TYPE SUBPROGRAM DEFINITIONS
      REAL FUNCTION PDOTT(A,B,C)
*NOTE THAT A,B,AND C REPRESENT P,Q,AND R RESPECTIVELY
      IMPLICIT REAL*4 (A-Z)
      COMMON /ARRAY1/DATAWRDR(400)
      EQUIVALENCE (DATAWRDR(57),IXMY)
     *,(DATAWRDR(60),IXY)
     *,(DATAWRDR(61),IXZ)
     *,(DATAWRDR(62),IYMZ)
     *,(DATAWRDR(63),IYOY)
     *,(DATAWRDR(65),IZMX)
     *,(DATAWRDR(66),IZOZ)
     *,(DATAWRDR(68),IFUNC)
     *,(DATAWRDR(81),LBODY)
     *,(DATAWRDR(87),MBODY)
     *,(DATAWRDR(88),NBODY)
      PDOTT=IFUNC*((LBODY-IXY*A*C+IXZ*A*B+IYMZ*B*C)
     1 +IYOY*(MBODY+IXY*B*C-IXZ*(A**2-C**2)+IZMX*A*C)
     1 +IZOZ*(NBODY-IXY*(B**2-A**2)-IXZ*B*C+IXMY*A*B))
      RETURN
      END
```

```
*
      REAL FUNCTION QDOTT(A,B,C,D)
*A,B,C,D REPRESENT P,Q,R,PDOT RESPECTIVELY
      IMPLICIT REAL*4 (A-Z)
      COMMON /ARRAY1/DATAWRDR(400)
      EQUIVALENCE (DATAWRDR(60),IXY)
     *,(DATAWRDR(61),IXZ)
     *,(DATAWRDR(64),IYY)
     *,(DATAWRDR(65),IZMX)
     *,(DATAWRDR(87),MBODY)
      QDOTT=(1/IYY)*(MBODY+IXY*(B*C+D)-IXZ*(A**2-C**2)+IZMX*A*C)
      RETURN
      END
*
      REAL FUNCTION RDOTT(A,B,C,D)
*A,B,C,D REPRESENT P,Q,R,PDOT RESPECTIVELY
      IMPLICIT REAL*4 (A-Z)
      COMMON /ARRAY1/DATAWRDR(400)
      EQUIVALENCE (DATAWRDR(57),IXMY)
     *,(DATAWRDR(60),IXY)
     *,(DATAWRDR(61),IXZ)
     *,(DATAWRDR(67),IZZ)
     *,(DATAWRDR(88),NBODY)
      RDOTT=(1/IZZ)*(NBODY-IXY*(B**2-A**2)-IXZ*(B*C-D)+IXMY*A*B)
      RETURN
      END
*
      REAL FUNCTION THETDOTT(A)
*A REPRESENTS PHI
      IMPLICIT REAL*4 (A-Z)
      COMMON /ARRAY1/DATAWRDR(400)
      EQUIVALENCE (DATAWRDR(106),QBODY),(DATAWRDR(114),RBODY)
      THETDOTT=QBODY*COS(A)-RBODY*SIN(A)
      RETURN
      END
*
      REAL FUNCTION PHIDOTT(A,B,C)
*A,B,C REPRESENT PHI,THETA,PSIDOT RESPECTIVELY
      IMPLICIT REAL*4 (A-Z)
      COMMON /ARRAY1/DATAWRDR(400)
      EQUIVALENCE (DATAWRDR(100),PBODY)
      PHIDOTT=PBODY+C*SIN(B)
      RETURN
      END
*
      REAL FUNCTION PSIDOTT(A,B)
*A,B REPRESENT PHI,THETA RESPECTIVELY
      IMPLICIT REAL*4 (A-Z)
      COMMON /ARRAY1/DATAWRDR(400)
      EQUIVALENCE (DATAWRDR(106),QBODY),(DATAWRDR(114),RBODY)
      PSIDOTT=(RBODY*COS(A)+QBODY*SIN(A))/COS(B)
      RETURN
      END
*
      REAL FUNCTION SPEED(A)
*A REPRESENTS AN UNNAMED FUNCTION/VARIABLR
      IMPLICIT REAL*4 (A-Z)
      IF(A.GE.1) GO TO 10
      STORE=1+A**2/4+A**4/40+A**6/1600
```

```
      GO TU 20
  10  STORE=1.839-.772/A**2+.164/A**4+.035/A**6
  20  SPEED=SQRT(STORE)
      RETURN
*
      END
**************************************************************************
*                                                                        *
*                     P O I N T     F U N C T I O N                      *
*                                                                        *
**************************************************************************
       SUBROUTINE POINTF(ARGVALUE,ARGLNTH,ARGPNT,ARFRAC,ARGLIST)
*    THIS IS THE FLOATING POINT VERSION OF POINI
C
C     SUBROUTINE PONT COMPARES THE FLOATING POINT VALUE IN ARGVALUE WITH
C     A LIST OF FLOATING POINT VALUES STARTIMG IN LOCATION ARGLIST
C     AND RETURNS ARGPNT AND ARFRAC TO THE M1IN PRO7RAM
C     WHERE ARGLIST IS A LIST OF (ARGLNTH) REAL NUMBERS IN ASCENDING ORDE
C     AND WHEFE ARGPNT IS THE POSIIION OF THE LARGEST VALUE IN ARGLIST TH
C     IS LESS THAN ARGVALUE AND ARFRAC IS A SCALED FRACTION VALUE REPRESE
C     THE DISTANCE BETWEEN THE TWO POINTS AR7PNT AND ARGPNTPLUS ONE
C
       INTEGER*2 ARGLNTH,ARGPNT
       REAL ARGVALUE,ARGLIST(ARGLNTH)
     2 IF (ARGPNT.LT.1) GOTO 150
     4 IF (ARGPNT.GT.ARGLNTH) ARGPNT=ARGLNTH
    10 IF (ARGVALUE-ARGLIST(ARGPNT)) 70,130,20
    20 ITEMP=ARGPNT+1
    30 IF (ITEMP.GT.ARGLNTH) GO TO 130
    40 IF(ARGVALUE-ARGLIST(ITEMP)) 100,120,50
    50 ARGPNT=ITEMP
    60 GO TO 20
    70 IF (ARGPNT-1) 150,130,80
    80 ARGPNT=ARGPNT-1
    90 GO TO 10
   100 ARFRAC=(ARGVALUE-ARGLIST(ARGPNT))/(ARGLIST(ITEMP)-ARGLIST(ARGPNT))
   110 GO TO 140
   120 ARGPNT=ITEMP
   130 ARFRAC=0
   132 IF(ARGPNT.LT.ARGLNTH) GOTO 140
   135 ARGPNT=ARGLNTH-1
   137 ARFRAC=1.0
   140 RETURN
   150 ARGPNT=1
   160 GO TO 10
       END
```

#   #   #   #   #   #   #   #   #   #   #   #   #   #   #   #   #   #   #

NOTE: Pages 136 to 148 inclusive contain all the aerodynamic coefficients used within this program.

CMO	63						
.0450	.0040	-.0286	-.0570	-.0745	-.0730	-.0720	-.0670
-.0630	.0270	.0012	-.0137	-.0250	-.0408	-.0607	-.0632
-.0660	-.0700	-.0045	-.0060	-.0092	-.0150	-.0230	-.0343
-.0490	-.0806	-.1290	-.0080	-.0070	-.0070	-.0110	-.0170
-.0250	-.0370	-.0613	-.1000	-.0125	-.0110	-.0087	-.0074
-.0080	-.0115	-.0210	-.0530	-.0820	-.0219	-.0140	-.0085
-.0041	-.0035	-.0068	-.0155	-.0450	-.0900	-.0180	-.0140
-.0080	-.0028	.0014	.0002	-.0100	-.0400	-.0810	
CME	378						
.1155	.1100	.1020	.0900	.0910	.0940	.0970	.1040
.1110	.0770	.0710	.0660	.0660	.0680	.0720	.0770
.0860	.0940	.0435	.0385	.0385	.0405	.0455	.0525
.0608	.0800	.1000	.0295	.0240	.0235	.0308	.0355
.0420	.0500	.0730	.1000	.0270	.0200	.0210	.0250
.0300	.0360	.0460	.0715	.0980	.0240	.0170	.0150
.0150	.0215	.0295	.0410	.0680	.0940	.0240	.0170
.0150	.0150	.0215	.0295	.0410	.0680	.0940	.0805
.0760	.0720	.0700	.0700	.0700	.0720	.0740	.0760
.0560	.0530	.0504	.0504	.0513	.0540	.0580	.0650
.0730	.0310	.0275	.0275	.0290	.0334	.0400	.0470
.0650	.0830	.0165	.0140	.0150	.0222	.0270	.0340
.0423	.0620	.0830	.0150	.0120	.0140	.0200	.0240
.0300	.0390	.0610	.0820	.0150	.0120	.0100	.0120
.0163	.0240	.0340	.0560	.0760	.0150	.0120	.0100
.0120	.0163	.0240	.0340	.0560	.0760	.0490	.0440
.0425	.0440	.0440	.0480	.0480	.0500	.0530	.0330
.0310	.0292	.0292	.0300	.0330	.0370	.0440	.0500
.0140	.0160	.0160	.0180	.0210	.0260	.0315	.0428
.0520	.0085	.0070	.0100	.0153	.0182	.0230	.0288
.0408	.0520	.0080	.0060	.0080	.0120	.0160	.0210
.0268	.0395	.0500	.0090	.0080	.0060	.0080	.0110
.0165	.0235	.0370	.0500	.0090	.0080	.0060	.0080
.0110	.0165	.0235	.0370	.0500	.0000	.0000	.0000
.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000
.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000
.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000
.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000
.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000
.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000
.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000
.0000	.0000	.0000	.0000	-.0480	-.0490	-.0495	-.0500
-.0538	-.0616	-.0690	-.0750	-.0800	-.0310	-.0310	-.0330
-.0354	-.0410	-.0460	-.0540	-.0640	-.0730	-.0140	-.0147
-.0158	-.0190	-.0250	-.0320	-.0400	-.0540	-.0655	-.0085
-.0120	-.0175	-.0234	-.0310	-.0380	-.0470	-.0630	-.0770
-.0070	-.0109	-.0150	-.0220	-.0290	-.0370	-.0445	-.0605
-.0740	-.0053	-.0082	-.0140	-.0210	-.0280	-.0350	-.0430
-.0580	-.0740	-.0053	-.0082	-.0140	-.0210	-.0280	-.0350
-.0430	-.0580	-.0740	-.0890	-.0920	-.0980	-.1025	-.1070
-.1190	-.1280	-.1400	-.1500	-.0600	-.0615	-.0700	-.0818
-.0935	-.1050	-.1140	-.1300	-.1400	-.0275	-.0352	-.0477
-.0590	-.0690	-.0765	-.0830	-.0940	-.1040	-.0227	-.0313
-.0438	-.0585	-.0720	-.0795	-.0852	-.0957	-.1050	-.0180
-.0260	-.0395	-.0560	-.0760	-.0925	-.1050	-.1180	-.1235
-.0120	-.0218	-.0358	-.0545	-.0760	-.0950	-.1110	-.1320
-.1380	-.0120	-.0218	-.0358	-.0545	-.0760	-.0950	-.1110
-.1320	-.1380						
CMDBF1	63						

-.00064	-.00077	-.00090	-.00095	-.00100	-.00105	-.00128	-.00188
-.00240	-.00034	-.00045	-.00055	-.00065	-.00075	-.00092	-.00109
-.00162	-.00209	-.00017	-.00022	-.00026	-.00039	-.00051	-.00073
-.00094	-.00138	-.00186	-.00013	-.00019	-.00025	-.00034	-.00042
-.00063	-.00084	-.00132	-.00179	-.00009	-.00015	-.00021	-.00029
-.00036	-.00057	-.00077	-.00128	-.00179	-.00009	-.00013	-.00017
-.00026	-.00034	-.00056	-.00077	-.00128	-.00179	-.00009	-.00013
-.00017	-.00026	-.00034	-.00056	-.00077	-.00128	-.00179	
CMDBF2	63						
-.00150	-.00167	-.00183	-.00200	-.00217	-.00234	-.00297	-.00387
-.00400	-.00109	-.00123	-.00136	-.00161	-.00186	-.00225	-.00263
-.00362	-.00375	-.00057	-.00076	-.00095	-.00128	-.00160	-.00204
-.00247	-.00327	-.00339	-.00025	-.00048	-.00070	-.00110	-.00150
-.00194	-.00237	-.00310	-.00324	-.00007	-.00033	-.00059	-.00101
-.00143	-.00188	-.00232	-.00304	-.00325	.00000	-.00023	-.00046
-.00086	-.00126	-.00180	-.00234	-.00303	-.00347	.00000	-.00023
-.00045	-.00083	-.00121	-.00177	-.00233	-.00304	-.00353	
CMDSB1	63						
.00090	.00084	.00078	.00070	.00067	.00058	.00049	.00040
.00041	.00073	.00066	.00058	.00053	.00046	.00041	.00034
.00027	.00020	.00050	.00044	.00038	.00033	.00028	.00022
.00016	.00010	.00004	.00029	.00026	.00022	.00018	.00014
.00011	.00007	.00002	.00000	.00024	.00021	.00017	.00013
.00008	.00008	.00007	.00000	.00000	.00015	.00012	.00008
.00005	.00002	.00001	.00000	.00000	.00000	.00015	.00012
.00008	.00005	.00002	.00001	.00000	.00000	.00000	
CMDSB2	63						
.00104	.00100	.00091	.00087	.00082	.00080	.00078	.00076
.00074	.00102	.00095	.00086	.00078	.00074	.00066	.00064
.00062	.00060	.00076	.00069	.00062	.00057	.00051	.00046
.00041	.00026	.00011	.00055	.00050	.00044	.00041	.00038
.00034	.00030	.00020	.00010	.00034	.00029	.00023	.00018
.00012	.00009	.00006	.00000	.00000	.00034	.00028	.00021
.00016	.00010	.00006	.00002	.00000	.00000	.00034	.00028
.00021	.00016	.00010	.00006	.00002	.00000	.00000	
CMQ	252						
-2.70	-2.52	-1.60	-0.08	-0.27	-3.66	-2.91	-0.66
-0.66	-2.00	-1.82	-0.75	-0.88	-0.67	-3.06	-1.86
1.34	3.34	-1.58	-1.49	-1.05	-1.34	-1.31	-2.73
-1.83	-0.33	1.07	-1.40	-1.40	-1.43	-1.50	-1.80
-2.10	-2.10	-1.80	-1.50	-1.55	-1.55	-1.55	-1.60
-1.64	-2.00	-2.90	-2.90	-2.00	-0.60	-1.00	-1.70
-2.40	-2.60	-2.30	-1.90	-1.40	-1.00	-0.60	-1.00
-1.70	-2.40	-2.60	-2.30	-1.90	-1.40	-1.00	-3.00
-3.00	-3.00	-1.00	-2.25	-3.00	-2.25	0.00	0.00
-2.30	-2.30	-2.15	-1.80	-2.65	-2.40	-1.20	2.00
4.00	-1.73	-1.73	-1.75	-1.80	-2.30	-2.40	-1.50
0.00	1.40	-1.40	-1.40	-1.43	-1.50	-1.80	-2.10
-2.10	-1.80	-1.50	-1.55	-1.55	-1.55	-1.60	-1.64
-2.00	-2.90	-2.90	-2.00	-0.60	-1.00	-1.70	-2.40
-2.60	-2.30	-1.90	-1.40	-1.00	-0.60	-1.00	-1.70
-2.40	-2.60	-2.30	-1.90	-1.40	-1.00	-3.36	-3.58
-4.70	-2.12	-4.65	-2.20	-1.45	0.80	0.80	-2.66
-2.88	-3.85	-2.92	-5.05	-1.60	-0.40	2.80	4.80
-1.91	-2.02	-2.60	-2.36	-3.50	-2.00	-1.10	0.40
1.80	-1.40	-1.40	-1.43	-1.50	-1.80	-2.10	-2.10
-1.80	-1.50	-1.55	-1.55	-1.55	-1.60	-1.64	-2.00
-2.90	-2.90	-2.00	-0.60	-1.00	-1.70	-2.40	-2.60
-2.30	-1.90	-1.40	-1.00	-0.60	-1.00	-1.70	-2.40

-2.60	-2.30	-1.90	-1.40	-1.00	-3.75	-4.20	-6.52
-3.32	-7.22	-1.34	-0.59	1.66	1.66	-3.05	-3.50
-5.67	-4.12	-7.62	-0.74	0.46	3.66	5.66	-2.11
-2.33	-3.51	-2.96	-4.78	-1.57	-0.67	0.83	2.23
-1.40	-1.40	-1.43	-1.50	-1.80	-2.10	-2.10	-1.80
-1.50	-1.55	-1.55	-1.55	-1.60	-1.64	-2.00	-2.90
-2.90	-2.00	-0.60	-1.00	-1.70	-2.40	-2.60	-2.30
-1.90	-1.40	-1.00	-0.60	-1.00	-1.70	-2.40	-2.60
-2.30	-1.90	-1.40	-1.00				
CNO	63						
-.0351	.2100	.4520	.6948	.9362	1.1500	1.3630	1.7880
2.2120	-.0475	.1535	.3475	.5500	.7578	.9690	1.2010
1.6660	2.1300	-.0472	.0761	.2280	.3879	.5730	.7610
.9710	1.4220	1.8730	-.0640	.0520	.1815	.3318	.5040
.6950	.9080	1.3450	1.7820	-.0625	.0360	.1535	.2958
.4690	.6520	.8580	1.3030	1.7600	-.0568	.0240	.1190
.2518	.4199	.6000	.8000	1.2350	1.7080	-.0568	.0240
.1130	.2429	.4079	.5910	.7920	1.2240	1.6900	
CNDE	63						
.0061	.0061	.0060	.0055	.0050	.0050	.0050	.0050
.0050	.0040	.0040	.0040	.0040	.0040	.0040	.0040
.0040	.0040	.0022	.0023	.0022	.0021	.0020	.0025
.0030	.0040	.0050	.0018	.0018	.0015	.0014	.0013
.0020	.0025	.0035	.0048	.0015	.0016	.0015	.0014
.0015	.0019	.0023	.0035	.0048	.0009	.0013	.0015
.0015	.0018	.0022	.0023	.0035	.0048	.0009	.0013
.0015	.0015	.0018	.0022	.0023	.0035	.0048	
CNDBF1	63						
.00068	.00077	.00085	.00091	.00097	.00111	.00132	.00188
.00308	.00030	.00037	.00044	.00056	.00068	.00090	.00115
.00167	.00299	.00017	.00025	.00032	.00044	.00056	.00080
.00103	.00158	.00282	.00013	.00020	.00027	.00040	.00052
.00074	.00096	.00158	.00267	.00009	.00016	.00023	.00037
.00051	.00069	.00087	.00158	.00256	.00008	.00013	.00017
.00034	.00050	.00067	.00083	.00158	.00248	.00008	.00013
.00017	.00031	.00045	.00064	.00083	.00158	.00248	
CNDBF2	63						
.00147	.00162	.00176	.00195	.00214	.00250	.00271	.00372
.00405	.00105	.00119	.00133	.00156	.00179	.00209	.00248
.00348	.00379	.00052	.00071	.00090	.00119	.00147	.00191
.00235	.00316	.00364	.00024	.00047	.00070	.00102	.00133
.00182	.00230	.00298	.00365	.00006	.00032	.00058	.00091
.00124	.00173	.00222	.00290	.00361	.00000	.00023	.00045
.00082	.00118	.00165	.00212	.00284	.00341	.00000	.00023
.00045	.00082	.00118	.00165	.00212	.00284	.00333	
CNDSB1	63						
-.00044	-.00036	-.00028	-.00018	-.00011	-.00007	-.00003	.00000
.00000	-.00033	-.00024	-.00016	-.00012	-.00004	-.00003	-.00003
.00000	.00000	-.00017	-.00012	-.00007	-.00005	-.00003	.00000
.00000	.00000	.00000	-.00010	-.00008	-.00005	-.00004	-.00003
.00000	.00000	.00000	.00000	-.00008	-.00007	-.00005	-.00004
-.00003	.00000	.00000	.00000	.00000	-.00006	-.00005	-.00003
-.00001	.00000	.00000	.00000	.00000	.00000	-.00006	-.00005
-.00003	-.00001	.00000	.00000	.00000	.00000	.00000	
CNDSB2	63						
-.00074	-.00064	-.00052	-.00041	-.00030	-.00022	-.00014	.00002
.00002	-.00056	-.00046	-.00036	-.00028	-.00024	-.00014	-.00008
.00004	.00004	-.00040	-.00034	-.00028	-.00020	-.00012	-.00008
-.00003	.00007	.00007	-.00035	-.00029	-.00022	-.00016	-.00009

-.00006	-.00003	.00000	.00000	-.00025	-.00021	-.00016	-.00012
-.00008	-.00006	-.00003	.00000	.00000	-.00017	-.00014	-.00010
-.00008	-.00005	-.00004	-.00002	.00000	.00000	-.00017	-.00014
-.00010	-.00008	-.00005	-.00004	-.00002	.00000	.00000	
CCO	63						
.1777	.1684	.1579	.1472	.1405	.1310	.1183	.0961
.0765	.1554	.1476	.1369	.1264	.1175	.1092	.0981
.0787	.0593	.1238	.1143	.1059	.0989	.0921	.0844
.0754	.0664	.0568	.1054	.0952	.0876	.0825	.0782
.0753	.0701	.0638	.0555	.0947	.0833	.0759	.0723
.0705	.0689	.0660	.0627	.0553	.0836	.0735	.0673
.0641	.0633	.0630	.0624	.0605	.0538	.0835	.0734
.0672	.0640	.0632	.0629	.0623	.0604	.0537	
CCE	378						
.066	.059	.046	.036	.026	.017	.008	-.009
-.022	.042	.034	.027	.022	.016	.009	.006
.000	-.001	.023	.019	.017	.012	.008	.006
.007	.002	.014	.022	.018	.013	.008	.004
.005	.005	.003	.015	.023	.017	.011	.008
.006	.005	.005	.005	.015	.019	.011	.003
.003	.003	.003	.004	.006	.016	.019	.011
.003	.003	.003	.003	.004	.006	.016	.026
.025	.021	.014	.010	.007	.003	-.004	-.010
.015	.013	.011	.007	.002	.000	.001	.003
.007	.009	.007	.005	.003	.001	.000	.001
-.001	.006	.008	.006	.003	-.001	-.004	-.003
.000	-.004	.003	.008	.004	.002	.001	.000
.000	-.002	-.004	.004	.007	.005	.002	.000
-.001	-.001	.002	.000	.009	.007	.005	.002
.000	-.001	-.001	.002	.000	.009	.008	.005
.004	.000	-.002	-.005	-.008	-.014	-.017	.004
.003	.001	.000	-.003	-.005	-.005	-.006	-.004
.003	.002	.002	-.002	-.003	-.004	-.004	-.003
.002	.001	.002	.001	-.001	-.004	-.005	-.004
-.007	-.002	.002	.000	.000	-.002	-.003	-.003
-.005	-.006	-.004	.003	.002	.002	.000	-.001
-.002	-.001	-.005	-.001	.003	.002	.002	.000
-.001	-.002	-.001	-.005	-.001	.000	.000	.000
.000	.000	.000	.000	.000	.000	.000	.000
.000	.000	.000	.000	.000	.000	.000	.000
.000	.000	.000	.000	.000	.000	.000	.000
.000	.000	.000	.000	.000	.000	.000	.000
.000	.000	.000	.000	.000	.000	.000	.000
.000	.000	.000	.000	.000	.000	.000	.000
.000	.000	.000	.000	.000	.000	.000	.000
.000	.000	.000	.000	.007	.009	.013	.015
.015	.023	.029	.037	.036	.005	.006	.009
.010	.012	.017	.022	.028	.028	.003	.005
.006	.009	.012	.014	.019	.025	.034	.002
.004	.005	.007	.009	.014	.017	.024	.032
.001	.003	.004	.007	.010	.012	.016	.023
.030	.000	.003	.004	.007	.009	.012	.015
.022	.034	.000	.003	.004	.007	.009	.012
.015	.022	.034	.035	.041	.043	.047	.057
.065	.076	.075	.067	.023	.027	.028	.038
.040	.053	.057	.064	.060	.010	.013	.014
.020	.031	.038	.047	.058	.075	.004	.010
.012	.017	.029	.043	.053	.063	.088	.008
.012	.013	.020	.029	.041	.056	.070	.097

.006	.011	.016	.024	.034	.048	.057	.070
.113	.006	.011	.016	.024	.034	.048	.057
.070	.113						
CCDBF1	63						
-.00024	-.00020	-.00015	-.00005	.00004	.00013	.00017	.00024
.00038	-.00009	-.00006	-.00003	.00003	.00009	.00015	.00021
.00028	.00042	.00000	.00003	.00007	.00009	.00010	.00012
.00013	.00019	.00032	.00000	.00000	.00000	.00000	.00005
.00008	.00010	.00016	.00026	.00000	.00000	.00000	.00000
.00000	.00003	.00005	.00012	.00022	.00000	.00000	.00000
.00000	.00000	.00002	.00004	.00009	.00017	.00000	.00000
.00000	.00000	.00000	.00002	.00004	.00009	.00017	
CCDBF2	63						
.00048	.00055	.00061	.00071	.00080	.00090	.00103	.00129
.00144	.00028	.00035	.00041	.00048	.00054	.00064	.00076
.00100	.00121	.00010	.00018	.00026	.00033	.00040	.00057
.00073	.00102	.00125	.00003	.00013	.00022	.00029	.00036
.00053	.00069	.00102	.00130	.00000	.00010	.00020	.00027
.00034	.00050	.00066	.00104	.00134	.00000	.00009	.00018
.00024	.00030	.00046	.00061	.00100	.00136	.00000	.00009
.00017	.00023	.00029	.00045	.00061	.00099	.00134	
CCDSB1	63						
.00075	.00070	.00066	.00054	.00048	.00039	.00026	.00009
.00000	.00042	.00042	.00042	.00040	.00036	.00033	.00029
.00021	.00014	.00031	.00030	.00028	.00026	.00023	.00019
.00014	.00007	.00001	.00024	.00021	.00018	.00014	.00011
.00009	.00007	.00001	.00000	.00016	.00012	.00009	.00007
.00004	.00003	.00001	.00000	.00000	.00011	.00010	.00008
.00006	.00003	.00002	.00000	.00000	.00000	.00011	.00010
.00008	.00006	.00003	.00002	.00000	.00000	.00000	
CCDSB2	63						
.00072	.00071	.00067	.00066	.00058	.00052	.00044	.00029
.00004	.00062	.00061	.00062	.00054	.00047	.00042	.00034
.00021	.00008	.00048	.00046	.00043	.00039	.00036	.00032
.00027	.00016	.00003	.00040	.00038	.00035	.00029	.00024
.00018	.00012	.00007	.00000	.00035	.00032	.00028	.00023
.00017	.00011	.00006	.00000	.00000	.00027	.00021	.00016
.00011	.00006	.00005	.00005	.00000	.00000	.00027	.00021
.00016	.00011	.00006	.00005	.00005	.00000	.00000	
CLB	252						
-.00141	-.00157	-.00161	-.00147	-.00140	-.00123	-.00117	-.00103
-.00092	-.00115	-.00117	-.00136	-.00140	-.00130	-.00101	-.00108
-.00095	-.00079	-.00074	-.00094	-.00104	-.00120	-.00126	-.00119
-.00132	-.00312	-.00342	-.00047	-.00081	-.00104	-.00125	-.00129
-.00118	-.00161	-.00296	-.00331	-.00032	-.00082	-.00109	-.00130
-.00136	-.00143	-.00170	-.00267	-.00310	-.00029	-.00053	-.00087
-.00106	-.00120	-.00145	-.00190	-.00254	-.00300	-.00029	-.00053
-.00087	-.00106	-.00120	-.00145	-.00190	-.00254	-.00300	-.00141
-.00157	-.00161	-.00147	-.00140	-.00123	-.00117	-.00103	-.00092
-.00115	-.00117	-.00136	-.00140	-.00130	-.00101	-.00108	-.00095
-.00079	-.00074	-.00094	-.00104	-.00120	-.00126	-.00119	-.00132
-.00312	-.00342	-.00047	-.00081	-.00104	-.00123	-.00129	-.00118
-.00161	-.00296	-.00331	-.00032	-.00082	-.00109	-.00130	-.00136
-.00143	-.00170	-.00267	-.00310	-.00029	-.00053	-.00087	-.00106
-.00120	-.00145	-.00190	-.00254	-.00300	-.00029	-.00053	-.00087
-.00106	-.00120	-.00145	-.00190	-.00254	-.00300	-.00135	-.00159
-.00160	-.00144	-.00135	-.00127	-.00117	-.00103	-.00092	-.00129
-.00121	-.00138	-.00142	-.00129	-.00100	-.00108	-.00095	-.00079
-.00086	-.00098	-.00120	-.00162	-.00168	-.00167	-.00180	-.00308

-.00332	-.00056	-.00084	-.00125	-.00147	-.00156	-.00163	-.00198
-.00264	-.00280	-.00040	-.00084	-.00118	-.00142	-.00162	-.00178
-.00206	-.00268	-.00272	-.00028	-.00054	-.00088	-.00106	-.00122
-.00146	-.00190	-.00256	-.00300	-.00028	-.00054	-.00088	-.00106
-.00122	-.00146	-.00190	-.00256	-.00300	-.00136	-.00160	-.00158
-.00144	-.00142	-.00152	-.00117	-.00103	-.00092	-.00129	-.00121
-.00135	-.00166	-.00184	-.00194	-.00108	-.00095	-.00079	-.00091
-.00106	-.00128	-.00152	-.00174	-.00193	-.00211	-.00256	-.00260
-.00057	-.00084	-.00113	-.00135	-.00157	-.00175	-.00201	-.00246
-.00256	-.00051	-.00080	-.00113	-.00134	-.00158	-.00181	-.00211
-.00252	-.00259	-.00028	-.00054	-.00088	-.00106	-.00122	-.00147
-.00191	-.00255	-.00299	-.00028	-.00054	-.00088	-.00106	-.00122
-.00147	-.00191	-.00255	-.00299				
CLBDE1	63						
.000011	.000006	.000016	.000006	.000000	.000000	.000000	.000000
-.000000	-.000005	-.000010	-.000010	-.000010	-.000010	-.000010	-.000010
-.000010	-.000010	-.000010	-.000010	-.000010	-.000010	-.000010	-.000011
-.000012	-.000014	-.000014	-.000010	-.000010	-.000010	-.000011	-.000013
-.000014	-.000015	-.000016	-.000016	-.000009	-.000009	-.000009	-.000010
-.000012	-.000012	-.000012	-.000012	-.000012	-.000012	-.000012	-.000012
-.000012	-.000012	-.000012	-.000012	-.000008	-.000008	-.000012	-.000012
-.000012	-.000012	-.000012	-.000012	-.000012	-.000008	-.000008	
CLBDE2	63						
-.000010	.000010	.000000	-.000016	-.000047	-.000047	-.000047	-.000047
-.000047	-.000010	-.000006	-.000006	-.000010	-.000017	-.000025	-.000025
-.000025	-.000025	-.000011	-.000011	-.000011	-.000010	-.000009	-.000017
-.000025	-.000041	-.000041	-.000010	-.000010	-.000010	-.000010	-.000011
-.000012	-.000013	-.000015	-.000015	-.000011	-.000011	-.000011	-.000012
-.000013	-.000012	-.000012	-.000011	-.000011	-.000010	-.000010	-.000010
-.000010	-.000010	-.000007	-.000005	-.000000	-.000000	-.000010	-.000010
-.000010	-.000010	-.000010	-.000007	-.000005	-.000000	-.000000	
CLBDSB1	63						
-.000023	-.000024	-.000025	-.000024	-.000023	-.000022	-.000021	-.000020
-.000019	-.000024	-.000025	-.000026	-.000025	-.000024	-.000023	-.000022
-.000021	-.000020	-.000014	-.000009	-.000003	.000001	.000004	.000007
.000010	.000013	.000016	-.000013	-.000008	-.000003	.000000	.000002
.000003	.000005	.000006	.000008	-.000013	-.000008	-.000003	.000000
.000000	.000000	.000000	.000000	.000000	-.000013	-.000008	-.000003
.000000	.000000	.000000	.000000	.000000	.000000	-.000013	-.000008
-.000003	.000000	.000000	.000000	.000000	.000000	.000000	
CLBDSB2	63						
-.000003	-.000003	-.000003	-.000004	-.000005	-.000005	-.000005	-.000005
-.000005	-.000005	-.000005	-.000005	-.000004	-.000003	-.000004	-.000005
-.000005	-.000005	-.000009	-.000009	-.000009	-.000006	-.000004	-.000003
-.000003	-.000003	-.000003	-.000007	-.000007	-.000008	-.000005	-.000003
-.000002	-.000001	-.000001	-.000001	-.000006	-.000006	-.000007	-.000004
-.000001	-.000001	0.000000	0.000000	0.000000	-.000006	-.000004	-.000003
-.000001	0.000000	0.000000	0.000000	0.000000	0.000000	-.000006	-.000004
-.000003	-.000001	0.000000	0.000000	0.000000	0.000000	0.000000	
CLBDSB3	63						
.000010	.000007	.000003	.000004	.000004	.000004	.000004	.000004
.000004	.000005	.000003	.000001	.000004	.000006	.000008	.000010
.000010	.000010	.000006	.000005	.000004	.000009	.000014	.000012
.000011	.000011	.000011	.000002	.000001	.000001	.000004	.000008
.000007	.000005	.000005	.000005	-.000003	-.000003	-.000003	-.000000
.000003	.000001	0.000000	0.000000	0.000000	-.000002	-.000002	-.000003
-.000001	0.000000	0.000000	0.000000	0.000000	0.000000	-.000002	-.000002
-.000003	-.000001	0.000000	0.000000	0.000000	0.000000	0.000000	
CLDA	378						

.00090	.00082	.00072	.00072	.00060	.00060	.00060	.00060
.00060	.00068	.00060	.00056	.00056	.00056	.00049	.00060
.00071	.00071	.00013	.00013	.00013	.00016	.00020	.00021
.00023	.00022	.00029	-.00003	-.00003	-.00003	-.00003	-.00003
.00002	.00008	.00000	.00006	-.00002	-.00002	-.00002	-.00002
-.00002	-.00002	-.00003	.00003	.00026	-.00004	-.00004	-.00004
-.00004	-.00004	-.00004	-.00004	.00004	.00027	-.00002	-.00002
-.00002	-.00002	-.00002	-.00001	.00000	.00007	.00027	.00134
.00115	.00105	.00105	.00093	.00093	.00093	.00093	.00093
.00095	.00081	.00075	.00073	.00075	.00072	.00081	.00090
.00090	.00040	.00040	.00040	.00043	.00047	.00051	.00055
.00043	.00051	.00026	.00024	.00022	.00026	.00030	.00036
.00043	.00062	.00080	.00022	.00022	.00022	.00022	.00022
.00028	.00035	.00056	.00086	.00021	.00021	.00021	.00021
.00021	.00028	.00036	.00059	.00085	.00020	.00020	.00020
.00020	.00020	.00027	.00035	.00057	.00084	.00151	.00144
.00137	.00137	.00139	.00139	.00139	.00139	.00139	.00100
.00100	.00095	.00090	.00095	.00105	.00110	.00115	.00115
.00052	.00053	.00055	.00061	.00068	.00076	.00084	.00112
.00135	.00042	.00038	.00035	.00043	.00052	.00065	.00079
.00110	.00140	.00040	.00037	.00034	.00039	.00045	.00060
.00076	.00110	.00132	.00043	.00039	.00035	.00039	.00043
.00058	.00073	.00108	.00132	.00041	.00037	.00033	.00037
.00041	.00058	.00076	.00108	.00130	.00101	.00101	.00101
.00101	.00127	.00127	.00127	.00127	.00127	.00096	.00096
.00100	.00110	.00127	.00142	.00156	.00170	.00170	.00044
.00053	.00062	.00076	.00091	.00108	.00124	.00167	.00200
.00025	.00035	.00046	.00062	.00078	.00101	.00124	.00163
.00195	.00024	.00033	.00042	.00059	.00076	.00100	.00124
.00166	.00189	.00026	.00033	.00041	.00058	.00075	.00100
.00125	.00163	.00189	.00027	.00033	.00040	.00057	.00074
.00100	.00126	.00163	.00190	.00179	.00179	.00179	.00179
.00205	.00225	.00225	.00225	.00225	.00111	.00111	.00119
.00142	.00174	.00208	.00240	.00272	.00272	.00062	.00072
.00083	.00114	.00146	.00179	.00212	.00256	.00290	.00039
.00058	.00078	.00111	.00145	.00178	.00211	.00246	.00270
.00035	.00056	.00078	.00110	.00143	.00175	.00208	.00242
.00268	.00036	.00056	.00076	.00111	.00146	.00178	.00210
.00242	.00266	.00036	.00057	.00078	.00112	.00146	.00178
.00210	.00245	.00266	.00200	.00200	.00200	.00200	.00217
.00156	.00156	.00156	.00156	.00137	.00137	.00152	.00184
.00240	.00290	.00347	.00404	.00404	.00090	.00102	.00115
.00171	.00227	.00274	.00321	.00373	.00410	.00068	.00101
.00135	.00206	.00278	.00317	.00356	.00400	.00425	.00052
.00091	.00130	.00207	.00285	.00323	.00362	.00408	.00440
.00050	.00091	.00133	.00209	.00285	.00323	.00362	.00382
.00412	.00052	.00090	.00128	.00207	.00287	.00323	.00360
.00390	.00416						

CLDR 252

.00046	.00043	.00039	.00037	.00034	.00031	.00029	.00024
.00019	.00033	.00031	.00028	.00025	.00022	.00019	.00016
.00010	.00004	.00024	.00022	.00019	.00017	.00014	.00012
.00009	.00003	.00000	.00019	.00016	.00013	.00011	.00009
.00007	.00005	.00001	.00000	.00016	.00013	.00010	.00008
.00006	.00004	.00002	.00000	.00000	.00014	.00011	.00008
.00006	.00004	.00003	.00001	.00000	.00000	.00014	.00011
.00008	.00006	.00004	.00003	.00001	.00000	.00000	.00067
.00063	.00058	.00053	.00048	.00043	.00038	.00028	.00018
.00052	.00048	.00044	.00039	.00034	.00028	.00021	.00008

.00000	.00038	.00034	.00030	.00026	.00022	.00016	.00010
.00003	.00000	.00030	.00026	.00022	.00019	.00016	.00010
.00005	.00001	.00000	.00026	.00022	.00018	.00015	.00012
.00007	.00003	.00000	.00000	.00020	.00016	.00012	.00009
.00007	.00004	.00001	.00000	.00000	.00020	.00016	.00012
.00009	.00007	.00004	.00001	.00000	.00000	.00062	.00057
.00052	.00051	.00049	.00047	.00045	.00041	.00037	.00050
.00046	.00041	.00039	.00036	.00032	.00028	.00020	.00012
.00040	.00035	.00031	.00026	.00021	.00015	.00010	.00003
.00000	.00032	.00028	.00024	.00019	.00014	.00010	.00006
.00001	.00000	.00026	.00023	.00019	.00014	.00010	.00006
.00003	.00000	.00000	.00020	.00016	.00012	.00008	.00004
.00002	.00001	.00000	.00000	.00020	.00016	.00012	.00008
.00004	.00002	.00001	.00000	.00000	.00030	.00027	.00025
.00026	.00027	.00023	.00020	.00013	.00006	.00028	.00025
.00022	.00022	.00022	.00017	.00011	.00004	.00000	.00026
.00023	.00020	.00018	.00015	.00011	.00008	.00002	.00000
.00024	.00021	.00018	.00015	.00012	.00008	.00005	.00000
.00000	.00022	.00019	.00016	.00013	.00009	.00006	.00003
.00000	.00000	.00020	.00016	.00012	.00009	.00006	.00003
.00000	.00000	.00000	.00020	.00016	.00012	.00009	.00006
.00003	.00000	.00000	.00000				

CLP 63

-.320	-.277	-.302	-.308	-.310	-.310	-.200	-.000
-.000	-.275	-.262	-.241	-.252	-.216	-.185	-.120
-.000	-.000	-.213	-.220	-.220	-.233	-.261	-.268
-.175	-.180	-.180	-.180	-.189	-.207	-.221	-.230
-.238	-.230	-.330	-.330	-.159	-.165	-.181	-.200
-.190	-.145	-.255	-.240	-.240	-.109	-.120	-.140
-.149	-.185	-.228	-.230	-.230	-.230	-.109	-.120
-.140	-.149	-.185	-.228	-.230	-.230	-.230	

CLR 63

.086	.112	.137	.145	.153	.161	.130	.145
.160	.064	.081	.098	.108	.118	.128	.104
.112	.124	.043	.049	.056	.064	.073	.071
.069	.075	.085	.031	.035	.040	.043	.046
.048	.051	.062	.070	.029	.032	.036	.041
.046	.048	.050	.059	.066	.026	.030	.035
.041	.046	.048	.050	.055	.065	.026	.030
.035	.041	.046	.048	.050	.055	.065	

CNB 252

.00110	.00040	.00020	-.00046	-.00198	-.00396	-.00481	-.00500
-.00500	.00122	.00039	-.00018	-.00112	-.00249	-.00432	-.00469
-.00500	-.00500	.00087	.00019	-.00022	-.00070	-.00220	-.00283
-.00300	-.00170	-.00185	.00066	.00027	-.00013	-.00047	-.00143
-.00196	-.00228	-.00178	-.00198	.00059	.00014	-.00016	-.00073
-.00121	-.00173	-.00180	-.00170	-.00170	-.00014	-.00016	-.00033
-.00112	-.00132	-.00130	-.00124	-.00136	-.00155	-.00014	-.00016
-.00033	-.00112	-.00132	-.00130	-.00124	-.00136	-.00155	.00110
.00040	.00020	-.00046	-.00198	-.00396	-.00481	-.00500	-.00500
.00122	.00039	-.00018	-.00112	-.00249	-.00432	-.00469	-.00500
-.00500	.00087	.00019	-.00022	-.00070	-.00220	-.00283	-.00300
-.00170	-.00185	.00066	.00027	-.00013	-.00047	-.00143	-.00196
-.00228	-.00178	-.00198	.00059	.00014	-.00016	-.00073	-.00121
-.00173	-.00180	-.00170	-.00170	-.00014	-.00016	-.00033	-.00112
-.00132	-.00130	-.00124	-.00136	-.00155	-.00014	-.00016	-.00033
-.00112	-.00132	-.00130	-.00124	-.00136	-.00155	.00122	.00056
.00022	-.00060	-.00230	-.00418	-.00481	-.00500	-.00500	.00126
.00057	-.00017	-.00121	-.00241	-.00305	-.00375	-.00427	-.00427

```
   .00132     .00061     .00010    -.00036    -.00096    -.00251    -.00259    -.00199
  -.00205     .00070     .00051     .00023    -.00036    -.00082    -.00163    -.00186
  -.00192    -.00198     .00069     .00034     .00012    -.00048    -.00102    -.00139
  -.00160    -.00192    -.00220    -.00014    -.00016    -.00033    -.00112    -.00132
  -.00130    -.00124    -.00136    -.00155    -.00014    -.00016    -.00033    -.00112
  -.00132    -.00130    -.00124    -.00136    -.00155     .00144     .00112     .00053
  -.00101    -.00218    -.00282    -.00421    -.00471    -.00500     .00139     .00074
  -.00057    -.00052    -.00131    -.00282    -.00279    -.00377    -.00427     .00096
   .00061    0.00014    -.00036    -.00128    -.00168    -.00222    -.00215    -.00247
   .00064     .00047     .00015    -.00055    -.00089    -.00123    -.00146    -.00190
  -.00218     .00068     .00044     .00016    -.00045    -.00102    -.00137    -.00156
  -.00197    -.00227    -.00014    -.00016    -.00033    -.00112    -.00132    -.00130
  -.00124    -.00136    -.00155    -.00014    -.00016    -.00033    -.00112    -.00132
  -.00130    -.00124    -.00136    -.00155
CNBDE1          63
  -.000021   -.000021   -.000021   -.000015   -.000005   -.000005   -.000005   -.000005
  -.000005   -.000004   -.000005   -.000004   -.000004   -.000000   -.000000   -.000000
  -.000000   -.000000   -.000000   -.000000   -.000000   -.000000   -.000000   -.000000
  -.000000   -.000000   -.000000   -.000000   -.000000   -.000000   -.000000   -.000000
  -.000000   -.000000   -.000000   -.000000   -.000000   -.000000   -.000000   -.000000
  -.000000   -.000000   -.000000   -.000000   -.000000   -.000000   -.000000   -.000000
  -.000000   -.000000   -.000000   -.000000   -.000000   -.000000   -.000000   -.000000
  -.000000   -.000000   -.000000   -.000000   -.000000   -.000000   -.000000
CNBDE2          63
  -.000006   -.000006   -.000010   -.000000   -.000030   -.000030   -.000030   -.000030
  -.000030   -.000000   -.000005   -.000000   -.000000   -.000000   -.000000   -.000000
  -.000000   -.000000   -.000000   -.000000   -.000000   -.000000   -.000000   -.000000
  -.000000   -.000000   -.000000   -.000000   -.000000   -.000000   -.000000   -.000000
  -.000000   -.000000   -.000000   -.000000   -.000000   -.000000   -.000000   -.000000
  -.000000   -.000000   -.000000   -.000000   -.000000   -.000000   -.000000   -.000000
  -.000000   -.000000   -.000000   -.000000   -.000000   -.000000   -.000000   -.000000
  -.000000   -.000000   -.000000   -.000000   -.000000   -.000000   -.000000
CNBDSB1         63
   .000021    .000021    .000021    .000027    .000018    .000014    .000010    .000002
   .000002    .000035    .000035    .000035    .000028    .000017    .000013    .000009
   .000001    .000001    .000032    .000032    .000032    .000026    .000000    .000000
   .000000    .000000    .000000    .000028    .000028    .000028    .000013    .000000
   .000000    .000000    .000000    .000000    .000024    .000024    .000024    .000000
   .000000    .000000    .000000    .000000    .000000    .000020    .000020    .000020
   .000000    .000000    .000000    .000000    .000000    .000000    .000020    .000020
   .000020    .000000    .000000    .000000    .000000    .000000    .000000
CNBDSB2         63
   .000006    .000006    .000006    .000006    .000006    .000006    .000007    .000007
   .000007    .000006    .000007    .000009    .000007    .000005    .000007    .000009
   .000009    .000009    .000011    .000013    .000015    .000010    .000006    .000005
   .000004    .000004    .000004    .000011    .000010    .000009    .000006    .000003
   .000002    .000001    .000001    .000001    .000011    .000009    .000006    .000004
   .000002    .000001   0.000000   0.000000   0.000000    .000011    .000007    .000004
   .000002   0.000000   0.000000   0.000000   0.000000   0.000000    .000011    .000007
   .000004    .000002   0.000000   0.000000   0.000000   0.000000   0.000000
CNBDSB3         63
  -.000021   -.000015   -.000009   -.000004    .000000    .000000   -.000002   -.000002
  -.000002   -.000008   -.000010   -.000013   -.000011   -.000009   -.000013   -.000018
  -.000018   -.000018   -.000005   -.000007   -.000008   -.000010   -.000017   -.000016
  -.000015   -.000018   -.000018    .000000    .000004    .000009   -.000001   -.000013
  -.000009   -.000008   -.000008   -.000008    .000004    .000013    .000024    .000010
  -.000005   -.000002   0.000000   0.000000   0.000000    .000002    .000007    .000012
   .000006   0.000000   0.000000   0.000000   0.000000   0.000000    .000002    .000007
   .000013    .000006   0.000000   0.000000   0.000000   0.000000   0.000000
```

CNDA	378						
.00136	.00113	.00090	.00067	.00044	.00012	-.00020	-.00084
-.00084	.00069	.00054	.00049	.00031	.00013	.00003	-.00008
-.00029	-.00029	.00053	.00043	.00031	.00023	.00015	.00006
-.00002	-.00014	-.00026	.00034	.00029	.00025	.00019	.00013
.00008	.00003	-.00006	-.00018	.00022	.00020	.00018	.00015
.00012	.00010	.00008	.00000	-.00011	.00008	.00005	.00003
.00003	.00003	.00001	-.00001	-.00001	-.00007	.00006	.00006
.00004	.00003	.00002	.00002	.00002	-.00002	-.00005	.00136
.00113	.00090	.00067	.00044	.00012	-.00020	-.00084	-.00084
.00058	.00048	.00036	.00020	.00004	-.00005	-.00014	-.00032
-.00032	.00038	.00027	.00019	.00012	.00005	-.00001	-.00007
-.00019	-.00027	.00025	.00020	.00015	.00010	.00005	.00000
-.00005	-.00015	-.00025	.00014	.00012	.00010	.00007	.00004
.00001	-.00002	-.00010	-.00023	.00006	.00004	.00002	.00002
.00002	.00000	-.00002	-.00009	-.00013	.00008	.00006	.00004
.00002	.00000	-.00001	-.00002	-.00008	-.00012	.00102	.00081
.00059	.00037	.00014	-.00010	-.00034	-.00082	-.00082	.00036
.00025	.00016	.00005	-.00006	-.00014	-.00022	-.00038	-.00038
.00021	.00012	.00006	.00000	-.00006	-.00011	-.00016	-.00026
-.00038	.00016	.00010	.00005	.00000	-.00004	-.00008	-.00012
-.00025	-.00035	.00008	.00004	.00001	-.00002	-.00005	-.00007
-.00009	-.00020	-.00034	.00006	.00003	.00000	-.00001	-.00002
-.00006	-.00010	-.00018	-.00026	.00004	.00002	.00000	-.00002
-.00003	-.00007	-.00010	-.00018	-.00025	.00018	.00007	-.00005
-.00018	-.00031	-.00049	-.00067	-.00103	-.00103	-.00001	-.00006
-.00013	-.00021	-.00028	-.00037	-.00046	-.00064	-.00064	-.00002
-.00006	-.00014	-.00020	-.00025	-.00031	-.00036	-.00059	-.00081
.00001	-.00004	-.00009	-.00016	-.00022	-.00029	-.00036	-.00053
-.00079	.00001	-.00005	-.00010	-.00016	-.00022	-.00028	-.00033
-.00052	-.00076	.00002	-.00004	-.00010	-.00013	-.00016	-.00023
-.00030	-.00051	-.00069	.00000	-.00005	-.00009	-.00012	-.00015
-.00023	-.00031	-.00054	-.00069	-.00054	-.00060	-.00066	-.00073
-.00080	-.00099	-.00118	-.00156	-.00156	-.00044	-.00044	-.00050
-.00064	-.00078	-.00098	-.00117	-.00156	-.00156	-.00028	-.00028
-.00041	-.00057	-.00073	-.00091	-.00108	-.00150	-.00190	-.00014
-.00024	-.00034	-.00051	-.00068	-.00087	-.00106	-.00143	-.00187
-.00010	-.00022	-.00034	-.00050	-.00066	-.00085	-.00104	-.00142
-.00177	-.00004	-.00019	-.00034	-.00049	-.00064	-.00083	-.00102
-.00142	-.00176	-.00006	-.00020	-.00034	-.00049	-.00064	-.00085
-.00105	-.00144	-.00176	-.00104	-.00111	-.00117	-.00124	-.00131
-.00149	-.00167	-.00203	-.00203	-.00088	-.00088	-.00088	-.00117
-.00146	-.00176	-.00206	-.00266	-.00266	-.00055	-.00052	-.00070
-.00103	-.00135	-.00178	-.00221	-.00266	-.00350	-.00027	-.00046
-.00064	-.00098	-.00132	-.00171	-.00210	-.00280	-.00332	-.00022
-.00042	-.00062	-.00094	-.00126	-.00167	-.00208	-.00253	-.00306
-.00010	-.00036	-.00062	-.00101	-.00140	-.00183	-.00226	-.00282
-.00350	-.00010	-.00038	-.00066	-.00105	-.00144	-.00187	-.00230
-.00284	-.00350						
CNDR	252						
-.00074	-.00071	-.00069	-.00064	-.00060	-.00053	-.00046	-.00018
.00000	-.00050	-.00048	-.00046	-.00041	-.00036	-.00029	-.00022
-.00009	.00000	-.00034	-.00032	-.00030	-.00027	-.00024	-.00018
-.00011	-.00002	.00000	-.00026	-.00026	-.00024	-.00019	-.00016
-.00011	-.00006	-.00001	.00000	-.00023	-.00021	-.00018	-.00014
-.00011	-.00007	-.00003	.00000	.00000	-.00020	-.00018	-.00014
-.00009	-.00004	.00000	.00000	.00000	.00000	-.00020	-.00018
-.00014	-.00009	-.00004	.00000	.00000	.00000	.00000	-.00118
-.00107	-.00096	-.00093	-.00090	-.00082	-.00074	-.00052	-.00031

-.00087	-.00079	-.00071	-.00068	-.00064	-.00053	-.00042	-.00021
.00000	-.00058	-.00054	-.00050	-.00046	-.00042	-.00034	-.00027
-.00010	.00000	-.00047	-.00043	-.00040	-.00035	-.00030	-.00023
-.00017	-.00006	.00000	-.00041	-.00037	-.00033	-.00026	-.00019
-.00014	-.00009	-.00002	.00000	-.00036	-.00030	-.00024	-.00015
-.00007	-.00004	-.00001	.00000	.00000	-.00036	-.00030	-.00024
-.00015	-.00007	-.00004	-.00001	.00000	.00000	-.00108	-.00102
-.00096	-.00096	-.00096	-.00090	-.00084	-.00072	-.00060	-.00092
-.00087	-.00081	-.00078	-.00076	-.00070	-.00064	-.00052	-.00040
-.00073	-.00068	-.00063	-.00056	-.00049	-.00040	-.00031	-.00007
.00000	-.00059	-.00054	-.00049	-.00042	-.00034	-.00025	-.00016
-.00002	.00000	-.00050	-.00044	-.00039	-.00031	-.00023	-.00015
-.00008	.00000	.00000	-.00036	-.00031	-.00025	-.00017	-.00008
-.00004	.00000	.00000	.00000	-.00035	-.00029	-.00022	-.00014
-.00005	-.00002	.00000	.00000	.00000	-.00040	-.00039	-.00038
-.00043	-.00047	-.00040	-.00032	-.00018	-.00004	-.00038	-.00036
-.00034	-.00036	-.00037	-.00030	-.00022	-.00008	.00000	-.00037
-.00033	-.00028	-.00025	-.00022	-.00018	-.00014	-.00004	.00000
-.00037	-.00032	-.00027	-.00022	-.00017	-.00013	-.00009	-.00001
.00000	-.00036	-.00031	-.00026	-.00020	-.00013	-.00010	-.00006
.00000	.00000	-.00036	-.00030	-.00024	-.00016	-.00007	-.00004
-.00001	.00000	.00000	-.00036	-.00029	-.00023	-.00015	-.00006
-.00003	.00000	.00000	.00000				

CNDRB 63

.000020	.000022	.000024	.000030	.000036	.000035	.000034	.000009
.000000	.000009	.000015	.000020	.000024	.000028	.000027	.000026
.000007	.000000	.000004	.000008	.000012	.000014	.000016	.000016
.000016	.000004	.000000	.000002	.000005	.000007	.000010	.000012
.000011	.000010	.000000	.000000	.000000	.000002	.000004	.000006
.000008	.000006	.000005	.000000	.000000	.000000	.000000	.000000
.000000	.000000	.000000	.000000	.000000	.000000	.000000	.000000
.000000	.000000	.000000	.000000	.000000	.000000	.000000	

CNP 63

.180	.149	.118	.086	.053	.026	.020	-.005
-.010	.128	.109	.090	.062	.033	.020	.006
-.010	-.012	.085	.069	.054	.032	.010	.001
-.008	-.015	-.017	.064	.049	.034	.015	-.005
-.010	-.016	-.018	-.021	.051	.036	.021	.004
-.013	-.015	-.018	-.021	-.024	.034	.020	.006
-.006	-.019	-.021	-.024	-.025	-.028	.030	.017
.005	-.007	-.020	-.022	-.024	-.024	-.028	

CNR 63

-0.460	-0.450	-0.385	-0.405	-0.620	-0.620	-0.620	-0.620
-0.620	-0.425	-0.450	-0.530	-0.700	-1.010	-1.390	-1.665
-1.730	-1.730	-0.340	-0.370	-0.420	-0.660	-0.910	-0.900
-0.645	-0.520	-0.520	-0.290	-0.310	-0.380	-0.550	-0.770
-0.985	-0.815	-0.580	-0.580	-0.220	-0.240	-0.300	-0.440
-0.620	-0.655	-0.490	-0.400	-0.400	-0.335	-0.240	-0.230
-0.240	-0.220	-0.250	-0.250	-0.250	-0.250	-0.335	-0.240
-0.230	-0.240	-0.220	-0.250	-0.250	-0.250	-0.250	

CYB 63

-.0182	-.0167	-.0171	-.0168	-.0135	-.0104	-.0088	-.0078
-.0078	-.0195	-.0180	-.0165	-.0145	-.0113	-.0100	-.0084
-.0070	-.0070	-.0182	-.0150	-.0135	-.0125	-.0115	-.0105
-.0095	-.0105	-.0090	-.0155	-.0140	-.0130	-.0118	-.0105
-.0095	-.0100	-.0100	-.0075	-.0140	-.0128	-.0120	-.0100
-.0085	-.0080	-.0080	-.0090	-.0075	-.0113	-.0110	-.0100
-.0080	-.0070	-.0068	-.0068	-.0068	-.0068	-.0113	-.0110
-.0100	-.0080	-.0070	-.0068	-.0068	-.0068	-.0068	

CYDA 378

-.00340	-.00329	-.00318	-.00284	-.00250	-.00025	-.00025	-.00025
-.00025	-.00222	-.00171	-.00121	-.00086	-.00051	-.00025	.00000
.00000	.00000	-.00031	-.00023	-.00015	-.00007	.00000	.00009
.00019	.00027	.00038	-.00028	-.00021	-.00014	-.00010	-.00006
-.00003	.00000	.00023	.00023	-.00010	-.00008	-.00006	-.00005
-.00003	-.00002	.00000	.00008	.00016	-.00011	-.00007	-.00004
-.00004	-.00004	-.00002	.00000	.00007	.00014	-.00015	-.00010
-.00006	-.00004	-.00002	.00000	.00002	.00008	.00014	-.00270
-.00222	-.00174	-.00118	-.00062	-.00012	-.00012	-.00012	-.00012
-.00081	-.00056	-.00031	-.00016	-.00002	.00008	.00019	.00019
.00019	-.00009	-.00008	-.00007	-.00003	.00001	.00004	.00008
.00017	.00017	-.00007	-.00005	-.00002	.00000	.00003	.00003
.00003	.00006	.00009	-.00008	-.00004	.00000	.00002	.00004
.00006	.00009	.00014	.00020	-.00009	-.00005	.00000	.00000
.00001	.00004	.00007	.00012	.00018	-.00011	-.00006	-.00001
.00000	.00001	.00004	.00007	.00013	.00019	-.00180	-.00119
-.00058	-.00040	-.00022	.00012	.00012	.00012	.00012	-.00042
-.00024	-.00006	.00006	.00018	.00020	.00022	.00022	.00022
.00000	.00000	.00000	.00002	.00005	.00007	.00010	.00021
.00021	-.00004	-.00001	.00002	.00005	.00009	.00011	.00013
.00018	.00018	-.00005	-.00001	.00003	.00006	.00009	.00011
.00013	.00017	.00022	-.00007	-.00003	.00002	.00004	.00006
.00008	.00012	.00017	.00022	-.00009	-.00005	.00000	.00002
.00005	.00008	.00011	.00016	.00022	-.00054	-.00036	-.00018
-.00010	-.00002	.00078	.00078	.00078	.00078	.00000	.00030
.00030	.00039	.00048	.00064	.00080	.00080	.00080	.00015
.00026	.00038	.00049	.00060	.00072	.00085	.00110	.00110
-.00001	.00009	.00020	.00025	.00030	.00036	.00044	.00060
.00060	-.00008	.00001	.00010	.00017	.00025	.00030	.00035
.00042	.00050	-.00010	-.00001	.00008	.00015	.00022	.00027
.00032	.00042	.00052	-.00012	-.00004	.00004	.00011	.00019
.00025	.00032	.00042	.00052	.00000	.00000	.00000	.00000
.00000	-.00001	-.00001	-.00001	-.00001	.00001	.00001	.00001
.00001	.00002	.00002	.00003	.00003	.00003	.00001	.00001
0.00001	.00000	.00000	.00005	.00011	.00069	.00069	-.00012
-.00003	.00006	.00011	.00016	.00021	.00026	.00040	.00040
-.00005	.00006	.00017	.00024	.00031	.00036	.00042	.00050
.00058	-.00004	.00005	.00016	.00025	.00035	.00038	.00041
.00050	.00060	-.00006	.00001	.00008	.00015	.00022	.00031
.00040	.00050	.00060	.00034	.00023	.00012	.00012	.00012
-.00168	-.00168	-.00168	-.00168	-.00020	-.00045	-.00071	-.00086
-.00100	-.00125	-.00151	-.00151	-.00151	-.00038	-.00052	-.00066
-.00107	-.00148	-.00124	-.00100	-.00005	-.00005	-.00060	-.00048
-.00036	-.00025	-.00014	-.00012	-.00010	-.00001	.00000	.00000
.00010	.00020	.00030	.00040	.00045	.00050	.00055	.00061
-.00001	.00009	.00019	.00029	.00039	.00045	.00052	.00061
.00070	.00000	.00005	.00011	.00014	.00028	.00038	.00048
.00058	.00068						

CYDR 63

.00205	.00190	.00175	.00175	.00174	.00170	.00165	.00155
.00145	.00160	.00150	.00139	.00134	.00129	.00122	.00114
.00099	.00084	.00112	.00106	.00099	.00090	.00081	.00069
.00056	.00017	.00000	.00083	.00077	.00070	.00058	.00045
.00034	.00023	.00006	.00000	.00068	.00062	.00055	.00041
.00026	.00018	.00010	.00002	.00000	.00060	.00051	.00043
.00029	.00015	.00007	.00000	.00000	.00000	.00060	.00051
.00043	.00029	.00015	.00007	.00000	.00000	.00000	

CYDRDSB1 7

.000002	.000004	.000006	.000001	.000000	-.000002	-.000002

CYDRDSB2 7

-.000029	-.000023	-.000012	-.000004	.000000	.000000	.000000

APPENDIX D - FLIGHT CONTROL SYSTEM PROGRAM LISTING

```
***************************************************************************
*                                                                         *
*                                                                         *
*                 S H U T T L E     D I G I T A L                         *
*                                                                         *
*          F L I G H T     C O N T R O L     S Y S T E M                  *
*                                                                         *
*                                                                         *
***************************************************************************
      SUBROUTINE SHTLFCS
      IMPLICIT REAL *4(A-Z)
      INTEGER *2 IAT,IPT,IDE,IGB
      LOGICAL*1 DATABYT
      COMMON /ARRAY1/DATAWRDR(400)
      COMMON /ARRAY3/DATABYT(17)
      COMMON /ARRAY4/ANGLE(3)
      LOGICAL*1 AUTOBF
      LOGICAL*1 IC,HOLD
      LOGICAL*1 TURNCORD
      EQUIVALENCE
     * (DATAWRDR(3),AY)
     *,(DATAWRDR(4),AZ)
     *,(DATAWRDR(9),BETA)
     *,(DATAWRDR(10),DSB)
     *,(DATAWRDR(34),DA)
     *,(DATAWRDR(35),DEL)
     *,(DATAWRDR(37),PRHCSOP)
     *,(DATAWRDR(38),DER)
     *,(DATAWRDR(40),DR)
     *,(DATAWRDR(43),DACM)
     *,(DATAWRDR(49),DE)
     *,(DATAWRDR(51),DBF)
     *,(DATAWRDR(54),DPJET)
     *,(DATAWRDR(55),DRJET)
     *,(DATAWRDR(72),RRHCSOP)
     *,(DATAWRDR(74),RPTASOP)
     *,(DATAWRDR(79),UZCMD)
     *,(DATAWRDR(80),UXCMD)
     *,(DATAWRDR(83),UYCMD)
     *,(DATAWRDR(84),MACH)
```

```
*,(DATAWRDR(100),PBODY)
*,(DATAWRDR(102),PE)
*,(DATAWRDR(103),QBAR)
*,(DATAWRDR(105),QDOT)
*,(DATAWRDR(106),QBODY)
*,(DATAWRDR(109),RDOT)
*,(DATAWRDR(112),SBHP)
*,(DATAWRDR(114),RBODY)
*,(DATAWRDR(124),UYGJET)
*,(DATAWRDR(131),TAS)
*,(DATAWRDR(132),PITCHPAN)
*,(DATAWRDR(138),XBAR)
*,(DATAWRDR(140),PITCHSTK)
*,(DATAWRDR(143),ROLLPAN)
*,(DATAWRDR(146),ROLLSTK)
*,(DATAWRDR(151),YAWTRIM)
*,(DATAWRDR(154),FLAPCMD)
*,(DATAWRDR(200),DEMAN)
*,(DATAWRDR(201),ESHAPE)
*,(DATAWRDR(202),GPS)
*,(DATAWRDR(203),DEMS)
*,(DATAWRDR(204),Q)
*,(DATAWRDR(205),PROD26)
*,(DATAWRDR(207),GJET)
*,(DATAWRDR(209),ETRIMIN)
*,(DATAWRDR(210),ETRIM)
*,(DATAWRDR(211),ELFBKIN)
*,(DATAWRDR(212),ELFBK)
*,(DATAWRDR(213),DETMPAN)
*,(DATAWRDR(214),DSBPC)
*,(DATAWRDR(215),DSBXTR)
*,(DATAWRDR(216),DSBXTRS)
*,(DATAWRDR(217),DETDSB)
*,(DATAWRDR(218),DETRIM)
*,(DATAWRDR(219),GDSB)
*,(DATAWRDR(220),GTRE)
*,(DATAWRDR(221),DETMRHC)
*,(DATAWRDR(222),DETR)
*,(DATAWRDR(223),DEMSP)
*,(DATAWRDR(224),RTPHI)
*,(DATAWRDR(225),RTANPHI)
*,(DATAWRDR(226),DJRTP)
*,(DATAWRDR(227),BCSL)
*,(DATAWRDR(228),DCSL)
*,(DATAWRDR(229),DQCT)
*,(DATAWRDR(230),DECMD)
*,(DATAWRDR(231),QFFIL)
*,(DATAWRDR(232),DCSQ)
*,(DATAWRDR(233),DCSLLO)
*,(DATAWRDR(234),DCSLHI)
*,(DATAWRDR(235),MIDPICK)
*,(DATAWRDR(236),KPIT)
*,(DATAWRDR(237),GDQ)
*,(DATAWRDR(238),DQLO)
*,(DATAWRDR(239),R)
*,(DATAWRDR(240),DQHI)
*,(DATAWRDR(241),DQHIA)
*,(DATAWRDR(242),DQHIN)
*,(DATAWRDR(243),NZA)
```

```
*,(DATAWRDR(244),NZP)
*,(DATAWRDR(245),DQLON)
*,(DATAWRDR(246),DQLOA)
*,(DATAWRDR(247),ALPHMIN)
*,(DATAWRDR(248),ALPHMAX)
*,(DATAWRDR(249),GTREDQCT)
*,(DATAWRDR(250),P)
*,(DATAWRDR(252),DATMRHC)
*,(DATAWRDR(253),PROD24)
*,(DATAWRDR(254),PROD25)
*,(DATAWRDR(257),PROD21)
*,(DATAWRDR(258),PROD22)
*,(DATAWRDR(259),PROD23)
*,(DATAWRDR(260),PROD27)
*,(DATAWRDR(261),SUM21)
*,(DATAWRDR(263),SUM23)
*,(DATAWRDR(264),DCSP)
*,(DATAWRDR(265),DACMD)
*,(DATAWRDR(266),DATSUMI)
*,(DATAWRDR(267),KDAMLIN)
*,(DATAWRDR(268),KDAMPAR)
*,(DATAWRDR(269),GRS)
*,(DATAWRDR(270),GDAC)
*,(DATAWRDR(271),GDA)
*,(DATAWRDR(272),DATSUM)
*,(DATAWRDR(273),PROD42)
*,(DATAWRDR(274),DATR)
*,(DATAWRDR(275),DRPRM)
*,(DATAWRDR(276),PSTAB)
*,(DATAWRDR(277),SUM24)
*,(DATAWRDR(278),DRPHI)
*,(DATAWRDR(279),PROD29)
*,(DATAWRDR(280),SUM22)
*,(DATAWRDR(281),GTX)
*,(DATAWRDR(282),YAWXFEED)
*,(DATAWRDR(283),PROD41)
*,(DATAWRDR(284),GP)
*,(DATAWRDR(285),GRH)
*,(DATAWRDR(288),SUM25)
*,(DATAWRDR(289),SUM25S)
*,(DATAWRDR(291),SUM26)
*,(DATAWRDR(292),PROD43)
*,(DATAWRDR(293),RSTAB)
*,(DATAWRDR(294),BETAG)
*,(DATAWRDR(295),BETAFILT)
*,(DATAWRDR(296),DATMPAN)
*,(DATAWRDR(297),DATP)
*,(DATAWRDR(299),PROD44)
*,(DATAWRDR(300),DRT)
*,(DATAWRDR(301),DRTMSF)
*,(DATAWRDR(302),DRMS)
*,(DATAWRDR(303),DRTMS)
*,(DATAWRDR(304),PROD1)
*,(DATAWRDR(305),SUM1)
*,(DATAWRDR(306),NYP)
*,(DATAWRDR(307),NYA)
*,(DATAWRDR(308),GRAY)
*,(DATAWRDR(309),PROD2)
*,(DATAWRDR(310),PROD3)
```

```
*,(DATAWRDR(311),SUM2)
*,(DATAWRDR(312),PROD5)
*,(DATAWRDR(313),DRCPF)
*,(DATAWRDR(314),KDRC)
*,(DATAWRDR(315),GDRC)
*,(DATAWRDR(316),GNYDRM)
*,(DATAWRDR(317),DRTRR)
*,(DATAWRDR(318),PROD7)
*,(DATAWRDR(319),GTRR)
*,(DATAWRDR(320),DRTRIM)
*,(DATAWRDR(321),SUM3)
*,(DATAWRDR(322),DRCMD)
*,(DATAWRDR(323),DACMC)
*,(DATAWRDR(324),GRXFD)
*,(DATAWRDR(325),GDRE)
*,(DATAWRDR(327),NR)
*,(DATABYT(10),HOLD)
*,(DATABYT(11),IC)
*,(DATABYT(13),TURNCORD)
*,(DATABYT(14),AUTOBF)
*,(ANGLE(1),ALFA)
*,(ANGLE(2),PHI)
*,(ANGLE(3),THETA)
 DIMENSION
*GPST(4),
*ALPHMINT(4),
*TALPHMAX(4,2),
*TKPIT(4,2),
*GTRET(4),
*GDSBT(4),
*KDAMLINT(4),
*KDAMPART(4),
*GRST(4),
*GDACT(4),
*GPT(4),
*GNYDRMT(4),
*GRAYT(4),
*GDRCT(4),
*GTRRT(4)
 DATA DEGRAD/57.29577951/
 DATA GPST/0.4,0.125,2.0,2.5/
 DATA ALPHMINT/-4.0,0.0,.6,.8/
 DATA TALPHMAX/20.0,15.0,15.0,30.0,.6,.8,1.5,3.0/
 DATA TKPIT/15.0,81.5,81.5,60.0,1.2,3.0,3.5,5.0/
 DATA GTRET/1.0,0.3,10.0,12.0/
 DATA GDSBT/0.1,0.25,1.5,3.0/
 DATA KDAMLINT/0.08,0.9324,0.6,1.5/
 DATA KDAMPART/0.0636,0.008,0.6,1.5/
 DATA GRST/1.0,0.25,0.6,1.5/
 DATA GDACT/0.45,2.5,1.15,3.0/
 DATA GPT/10.0,1.5,10.0,45.0/
 DATA GNYDRMT/0.05,0.015,250.0,900.0/
 DATA GRAYT/10.0,3.2,2.0,3.0/
 DATA GDRCT/1200.0,3800.0,1.15,5.0/
 DATA GTRRT/0.2,1.0,2.0,5.0/
 DATA DAMAX/20.0/
 DATA PRHCDB/1.15/
 DATA GPRHCT/0.3/
 DATA RPTADB/1.125/
```

```
      DATA NQLO/2.0/
      DATA NQHI/4.0/
      DATA DPCUTOFF/1.0/
      DATA DPJET1/0.0/
      DATA OLDPE/0.0/
      DATA DRJET1/0.0/
      DATA QBDB/0.25/
      DATA QADB/0.2/
      DATA TGJET12/0.5/
      DATA TGJET21/5.0/
      DATA GRPANTL/0.5/
      DATA GRPANTE/-0.5/
      DATA GRRHCTE/0.6/
      DATA GRRHCTL/0.2/
      DATA GQMWGP/1.5/
      DATA GQA/1.0/
      DATA NZMAX/3.75/
      DATA GQNL/4.0/
      DATA NZMIN/-1.0/
      DATA GSBHI/25.0/
      DATA GSBLO/10.0/
      DATA GPPANT/2.0/
      DATA GPP/1.0/
      DATA GTRA/0.2/
      DATA G/32.174/
      DATA RPTADB/1.125/
      DATA GDRTI/1.0/
      DATA YBBFILT/1.0/
      DATA DBFADB/0.5/
      DATA DBFBDB/1.0/
      DATA KDRMLIN/0.0131/
      DATA KDRMPAR /0.042/
      DATA DBFHA/1.0/
      DATA DBFHB/2.0/
      DATA DFRADB/0.9/
      DATA DFRBDB/1.0/
      DATA PEBDB/0.2/
      DATA PEADB/0.1/
      DATA NR1/1.0/
      DATA NR2/2.0/
      DATA NR3/3.0/
      DATA NR4/4.0/
      DATA GRCSA/-2.0/
      DATA DBFDCS1    /0.0/
      DATA DBFDC /0.0/
      DATA OLD25/0.0/
      DATA DSBC1/0.0/
      DATA RTANPHIN/0.0/
      DATA BCSLN/0.0/
      DATA QFFILN/0.0/
      DATA NZPN1,NZPN2/0.0,0.0/
      DATA PROD22N/0.0/
      DATA PROD29N/0.0/
      DATA BETAN/0.0/
      DATA NYPN1,NYPN2/0.0,0.0/
      DATA PROD2N/0.0/
      IF(IC) GO TO 1000
    3 CONTINUE
      NY=AY/32.174
```

```
      NZ=AZ/32.174
      ALPHA=ALFA*DEGRAD
      CGPOS=1293.36*XBAR+236.0
      P=PBODY*DEGRAD
      Q=QBODY*DEGRAD
      R=RBODY*DEGRAD
      SINALF=SIN(ALFA)
      SINALF1=SINALF
      IF(SINALF.LT.0.087155) SINALF1=0.087155
      COSALF=COS(ALFA)
      SINPHI=SIN(PHI)
      COSPHI=COS(PHI)
      COSTHETA=COS(THETA)
      DETMPAN=0.0
      IF(PITCHPAN.GE.0.5)DETMPAN=1.0
      IF(PITCHPAN.LE.-0.5)DETMPAN=-1.0
      DETMRHC=0.0
      IF(PITCHSTK.GE.0.5)DETMRHC=1.0
      IF(PITCHSTK.LE.-0.5)DETMRHC=-1.0
      DATMPAN=0.0
      IF(ROLLPAN.GE.0.5)DATMPAN=1.0
      IF(ROLLPAN.LE.-0.5)DATMPAN=-1.0
      DATMRHC=0.0
      IF(ROLLSTK.GE.0.5)DATMRHC=1.0
      IF(ROLLSTK.LE.-0.5)DATMRHC=-1.0
      DRT=0.0
      IF(YAWTRIM.GE.0.5)DRT=1.0
      IF(YAWTRIM.LE.-0.5)DRT=-1.0
      DBFMAN=0.0
      IF(FLAPCMD.GE.0.5)DBFMAN=1.0
      IF(FLAPCMD.LE.-0.5)DBFMAN=-1.0
*
* THE FOLLOWING SECTION CALCULATES ALL TABLE LOOKUP FUNCTION VALUES
*
      GPS=FUNCTION(MACH,GPST)
      GJET=0.0
      IF(QBAR.GE.TGJET12)GJET=TGJET21
      CALL POINTF(MACH,4,IAT,FRAC,TALPHMAX(1,2))
      ALPHMAX=TALPHMAX(IAT,1)+(TALPHMAX(IAT+1,1)-TALPHMAX(IAT,1))*FRAC
      ALPHMIN=FUNCTION(MACH,ALPHMINT)
      CALL POINTF(MACH,4,IPT,FRAC,TKPIT(1,2))
      KPIT=TKPIT(IPT,1)+(TKPIT(IPT+1,1)-TKPIT(IPT,1))*FRAC
      GTRE=FUNCTION(MACH,GTRET)
      GDSB=FUNCTION(MACH,GDSBT)
      KDAMPAR=FUNCTION(MACH,KDAMPART)
      KDAMLIN=FUNCTION(MACH,KDAMLINT)
      GRS=FUNCTION(MACH,GRST)
      GDAC=FUNCTION(MACH,GDACT)
      GP=FUNCTION(QBAR,GPT)
      GNYDRM=FUNCTION(TAS,GNYDRMT)
      GRAY=FUNCTION(MACH,GRAYT)
      KDRC=FUNCTION(MACH,GDRCT)
      GTRR=FUNCTION(MACH,GTRRT)
*
*   PITCH FCS PROCESSING
*
* DEMSP
      DEMAN=0
      IF(PRHCSOP.LT.-PRHCDB)DEMAN=PRHCSOP+PRHCDB
```

```
      IF(PRHCSOP.GT.PRHCDB)DEMAN=PRHCSOP-PRHCDB
      ESHAPE=(0.36+0.0484*ABS(DEMAN))*DEMAN
      IF(ESHAPE.GT.23.0)ESHAPE=23.0
      IF(ESHAPE.LT.-23.0)ESHAPE=-23.0
      DEMS=ESHAPE*GPS
      DEMSP=DEMS
      IF((MACH.GT.1.5).AND.(PRHCSOP.GT.19.5)) DEMSP=DEMS*2.0
*DETR
   20 DETMGPR=DETMRHC*GPRHCT
      IF(IC.OR.HOLD) GO TO 29
      DETR=FILT1(DETMGPR,.04,0.0,-1.0,DETRN)
      IF(DETR.GT.1.5) DETR=1.5
      IF(DETR.LT.-1.5) DETR=-1.5
   29 CONTINUE
*DPJET
      DPJET=DEMSP-Q
*UYCMD
      ABSDPJET=ABS(DPJET)
      NQ=NQHI
      IF(ABSDPJET.LE.DPCUTOFF) NQ=NQLO
      CALL HYSTER(QADB,QBDB,DPJET1,DPJET,OUTDPJET)
      DPJET1=DPJET
      UYCMD=NQ*OUTDPJET
      IF((MACH.LE.1.5).OR.(QBAR.GT.20.0))UYCMD=0.0
*UYGJET
   50 UYGJET=GJET*UYCMD
*RTANPHI
      TANPHI=SINPHI/COSPHI
      IF(TANPHI.GT.1.0)TANPHI=1.0
      IF(TANPHI.LT.-1.0)TANPHI=-1.0
   60 RTPHI=R*TANPHI
      IF(HOLD) GO TO 62
      RTANPHI=FILT1(RTPHI,.01961,.01961,-.9608,RTANPHIN)
   62 CONTINUE
*BCSL
      IF(MACH.GT.1.5) RTANPHI=0.0
   70 DJRTP=-DPJET-RTANPHI-DETR
      IF(HOLD) GO TO 72
      BCSL=FILT1(DJRTP,1.9615,-1.8846,-.9231,BCSLN)
   72 CONTINUE
*DCSLLO
      IF(HOLD) GO TO 74
      QFFIL=FILT1(Q,.02988,.02988,-.992,QFFILN)
   74 CONTINUE
      DCSQ=QFFIL*GQMWGP
      DQLOA=(ALPHA-ALPHMAX)*GQA
      NZA=NZ+1.8649*QDOT
      IF(HOLD) GO TO 76
      NZP=FILT2(NZA,.06475,.1295,.06475,-1.3094,.5683,NZPN1,NZPN2)
   76 CONTINUE
      DQLON=(NZP-NZMAX)*GQNL
      DQLO=DQLOA
      IF(DQLON.GT.DQLOA) DQLO=DQLON
   80 DCSLLO=DQLO+DCSQ
*DCSLHI
      DQHIA=(ALPHA-ALPHMIN)*GQA
      DQHIN=(NZP-NZMIN)*GQNL
      DQHI=DQHIN
      IF(DQHIA.LE.DQHIN)DQHI=DQHIA
```

```
   90 DCSLHI=DQHI+DCSW
*MIDPICK
      IF(DCSLLO.GE.DCSLHI) GO TO 98
      IF(DCSLHI.LT.BCSL) GO TO 91
      GO TO 93
   91 IF(DCSLHI.GT.DCSLLO) GO TO 97
      GO TO 98
   93 IF(BCSL.GE.DCSLLO) GO TO 98
      GO TO 99
   97 MIDPICK=DCSLHI
      GO TO 100
   98 MIDPICK=BCSL
      GO TO 100
   99 MIDPICK=DCSLLO
*DCSL
  100 IF(MACH.GT.1.5)GO TO 105
      DCSL=MIDPICK
      GO TO 110
  105 IF(QBAR.GT.2.0) GO TO 106
      DCSL=0.0
      GO TO 110
  106 DCSL=BCSL
*DQCT
  110 CONTINUE
      GDQ=KPIT/SQRT(QBAR+4.0)
      IF(GDQ.GE.6.0)GDQ=6.0
      IF(GDQ.LE.0.2)GDQ=0.2
      DQCT=DCSL*GDQ
*GTREDQCT
      GTREDQCT=GTRE*DQCT
*DSBXTRS
      DSBXTR=DSBPC*GDSB
      IF(DSB.GE.30.0) GO TO 115
      DSBXTRS=DSBXTR*GSBLO
      GO TO 120
  115 DSBXTRS=DSBXTR*GSBHI
*DETP
  120 DETP=DETMPAN*GPPANT
      DETDSB=DSBXTRS-DETP
*ETRIM
      ETRIMIN=GTREDQCT-DETP-UYGJET
      IF(IC.OR.HOLD) GO TO 121
      ETRIM=FILT1(ETRIMIN,.04,0.0,-1.0,ETRIMN)
  121 CONTINUE
      IF(ETRIM.GE.20.0)ETRIM=20.0
      IF (ETRIM.LE.-35.0)ETRIM=-35.0
*ELFBK
      ELFBKIN=(DETDSB/GTRE)+DE
      IF(IC.OR.HOLD) GO TO 122
      TEMP=50.0/GTRE
      ELFBK=FILT1(ELFBKIN,1.0/(TEMP+1.0),1.0/(TEMP+1.0),
     * (-TEMP+1.0)/(TEMP+1.0),ELFBKN)
  122 CONTINUE
*DETRIM
      IF(MACH.LE.12.0) GO TO 125
      DETRIM=ETRIM
      GO TO 130
  125 DETRIM=ELFBK
  130 CONTINUE
```

```
*DECMD
  140 DECMD=DETRIM+DQCT
      IF(DECMD.GT.20.0) DECMD=20.0
      IF(DECMD.LT.-35.0) DECMD=-35.0
*
*   END OF PITCH PROCESSING
*
*   ROLL FCS PROCESSING,PART 1
*
* DAMAN
      DAMAN=0.0
      IF(RRHCSOP.LT.-1.15) DAMAN=RRHCSOP+1.15
      IF(RRHCSOP.GT.1.15) DAMAN=RRHCSOP-1.15
*   PROD21
      PROD21=DAMAN*(KDAMLIN+KDAMPAR*ABS(DAMAN))
      IF(PROD21.GT.DAMAX)PROD21=DAMAX
      IF(PROD21.LT.-DAMAX)PROD21=-DAMAX
      IF(HOLD) GO TO 155
      PROD22=FILT1(PROD21,.0909,.0909,-.8182,PROD22N)
  155 CONTINUE
*GRS
      PROD23=GRS*PROD22
      IF(MACH.GT.1.5) PROD24=DATMRHC*.6
      IF(MACH.LE.1.5) PROD24=DATMRHC*.2
      IF(IC.OR.HOLD) GO TO 171
      PROD25=FILT1(PROD24,.04,0,-1.0,PROD25N)
  171 CONTINUE
      IF(PROD25.GT.2.5)PROD25=2.5
      IF(PROD25.LT.-2.5) PROD25=-2.5
      SUM21=PROD25+PROD23
      PROD27=SUM21*2.0
      IF(PROD27.GT.10.0)PROD27=10.0
      IF(PROD27.LT.-10.0)PROD27=-10.0
      PROD26=PROD27*SINALF1
* DRPHI
      DRPHI=57.3*G*SINPHI*COSTHETA/TAS
      IF(TURNCORD)GOTO 174
      DRPHI=0
* DRPRM
  174 DRPRM=R-DRPHI
      IF(HOLD) GO TO 178
      PROD29=FILT1(DRPRM,.9804,-.9804,-.9608,PROD29N)
  178 CONTINUE
* GRH
      GRH=28.8/(QBAR+10.0)
      IF(GRH.LT..05) GRH=.05
      IF(HOLD) GO TO 185
      BETAFILT=FILT1(BETA,.999,-.999,-.998,BETAN)
  185 CONTINUE
      BETAQ=BETAFILT
      IF(QBAR.LE.2.0)BETAQ=BETA
      IF(QBAR.GT.20.0)BETAQ=0.0
      SUM22=COSALF*BETAQ-PROD29*GRH-2*DRPRM
      SUM23=PROD26+SUM22
* RSTAB
      RSTAB=DRPRM*COSALF-P*SINALF
* PSTAB
      PSTAB=DRPRM*SINALF+P*COSALF
      PSTABG=PSTAB*GPP
```

```
      SUM24=SUM21-PSTABG
      DACM=SUM24*GDAC
*
*   END OF ROLL,PART 1 PROCESSING
*
*   YAW FCS PROCESSING
*
* DRMAN
      DRMAN=0.0
      IF(RPTASOP.LT.-1.125) DRMAN=RPTASOP+1.125
      IF(RPTASOP.GT.1.125) DRMAN=RPTASOP-1.125
* DRMS
  220 DRMS=(KDRMLIN+KDRMPAR*ABS(DRMAN))*DRMAN
      IF(DRMS .GT.  22.5) DRMS=22.5
      IF(DRMS .LT. -22.5) DRMS=-22.5
* DRTMS
      DRTGDR=DRT*GDRTI
      IF(IC.OR.HOLD) GO TO 221
      DRTMSF=FILT1(DRTGDR,.04,0,-1.0,DRTMSFN)
  221 CONTINUE
      IF(DRTMSF.LT.-2.0)DRTMSF=-2.0
      IF(DRTMSF.GT.2.0)DRTMSF=2.0
      DRTMS=DRTMSF+DRMS
*    SUM1
      PROD1= GNYDRM*DRTMS
      IF(TURNCORD) GO TO 225
      SUM1=-PROD1
      GOTO230
  225 NYA=NY+RDOT*1.8649
      IF(HOLD) GO TO 232
      NYP=FILT2(NYA,.06475,.1295,.06475,-1.309,.5683,NYPN1,NYPN2)
      SUM1=NYP-PROD1
  230 PROD2=FILT1(SUM1,.0909,.0909,-.8182,PROD2N)
  232 CONTINUE
*   PROD3
      PROD3=GRAY*PROD2
*   SUM2
      SUM2=PROD3+RSTAB
* GDRC
      GDRC=KDRC/(QBAR+80)
      IF(GDRC.GT.15.0)GDRC=15.0
      IF(GDRC.LT.1.2)GDRC=1.2
  234 PROD5=SUM2*GDRC
      IF(MACH .LE. 1.5) GOTO 235
* GDRE(EARLY)
      GDRE=-600.0/QBAR
      IF(GDRE.GT.-1.0)GDRE=-1.0
      IF(GDRE.LT.-6.0)GDRE=-6.0
      PROD7=GDRE*SUM23
      DRCPF=YBBFILT*PROD7
      IF(MACH .LE. 5.0) GOTO 236
      PROD7=0.0
      DRCPF=0.0
      GOTO 236
  235 DRCPF=YBBFILT*PROD5
  236 CONTINUE
* DRTRR
      DRTRR=DRCPF*GTRR
      IF(IC.OR.HOLD) GO TO 244
```

```
      DRTRIM=FILT1(DRTRR,.04,0,-1.0,DRTRIMN)
  244 CONTINUE
      IF(DRTRIM.GT.9.0)DRTRIM=9.0
      IF(DRTRIM.LT.-9.0)DRTRIM=-9.0
      GRXFD=150/QBAR
      IF(GRXFD.GT.1.0) GRXFD=1.0
      IF(GRXFD.LT..1) GRXFD=.1
      DACMC=0
      IF((MACH.LE.1.5).AND.(TURNCORD)) DACMC=GRXFD*DACM
      SUM3=DRCPF+DRTRIM-DACMC
* DRCMD
      IF(SUM3.GT.22.8) SUM3=22.8
      IF(SUM3.LT.-22.8) SUM3=-22.8
      DRCMD=SUM3
      DRJET=SUM23
      IF(MACH.GT.1.5) GO TO 262
* DRJET
      DRJET=0
      IF(MACH.GT.1.0) DRJET=GRCSA*SUM2
* NR
* YA
* YB
  262 ABSDRJET=ABS(DRJET)
      IF(QBAR.LE.20) GO TO 263
      NR=NR1
      IF(ABSDRJET.GT..5) NR=NR2
      IF(ABSDRJET.GT.1.0) NR=NR3
      IF(ABSDRJET.GT.1.5) NR=NR4
      GO TO 264
  263 NR=NR1
      IF(ABSDRJET.GT..5) NR=NR2
  264 YADB=.1
      YBDB=.2
      IF(MACH.GT.1.5) GO TO 265
      YADB=.5
      YBDB=.8
  265 CALL HYSTER(YADB,YBDB,DRJET1,DRJET,OUTDRJ)
      DRJET1=DRJET
      UZCMD=NR*OUTDRJ
*
*   END OF YAW PROCESSING
*
*   ROLL FCS PROCESSING,PART 2
*
*   PE
      COTALF=COSALF/SINALF
      IF(COTALF.GT.11.43) COTALF=11.43
      IF(COTALF.LT.-11.43) COTALF=-11.43
      PE=DRPRM*COTALF-BETAQ*SINALF-P
      ABSPE=ABS(PE)
*   YAWXFEED
      CALL HYSTER(PEADB,PEBDB,OLDPE,PE,PEOUT)
      OLDPE=PE
      YAWXFEED=PEOUT*ABSPE
      PROD41=0
      IF(QBAR.GT.2) PROD41=GP*YAWXFEED
*   DCSP
      IF(MACH.GT.1.5) GO TO 270
      GDA=200/(QBAR+80)
```

```
      IF(GDA.LT..1) GDA=.1
      IF(GDA.GT.(1.2/GDAC)) GDA=1.2/GDAC
      DCSP=GDA*DACM
      GO TO 275
  270 CONTINUE
      GDA=150/QBAR
      IF(GDA.LT..1) GDA=.1
      IF(GDA.GT.1) GDA=1
      DCSP=GDA*PROD41
  275 CONTINUE
      PROD42=DCSP*GTRA
*    UXCMD
      NP=2
      IF(ABSPE.GT.1) NP=4
      UXCMD=0
      IF((QBAR.LE.10).AND.(MACH.GT.1.5)) UXCMD=NP*PEOUT
*    PROD43
      SUM25=SIGN(1.0,UXCMD)-SIGN(1.0,UZCMD)
      SUM25S=SIGN(1.0,SUM25)
      ABS25=ABS(SUM25S)
      SUM26=SUM25-OLD25
      OLD25=SUM25S
      PROD43=SIGN(1.0,SUM26)*ABS25
*    GTX,PROD44
      GTX=1.25
      IF(MACH.GT.10) GTX=50/(QBAR+10)
      IF(GTX.LT.1.25) GTX=1.25
      IF(GTX.GT.5) GTX=5
      PROD44=PROD43*GTX
*    DATR
      DATR=0
      IF(MACH.LE.1.5) DATR=PROD42
      IF((MACH.GT.5).AND.(QBAR.GT.2)) DATR=PROD44
*    DATP
      DATP=DATMPAN*.5
      IF(MACH.GT.1.5) DATP=-.5*DATMPAN
*    DATSUMI
      DATSUM=DATR+DATP
      IF(IC.OR.HOLD) GO TO 300
      DATSUMI=FILT1(DATSUM,.04,0,-1.0,DATSUMN)
  300 CONTINUE
*    DACMD
      DACMD=DCSP+DATSUMI
      IF(DACMD.GT.10) DACMD=10
      IF(DACMD.LT.-10) DACMD=-10
*
*    END OF ROLL PROCESSING
*
*
*    ACTUATOR PROCESSING FOLLOWS
*
* DECDRR
      DECDRR=((DECMD-DACMD)-DER)*20.0
      IF(DECDRR.GE.20.0)DECDRR=20.0
      IF(DECDRR.LE.-20.0)DECDRR=-20.0
* DER
  430 DER=.04*DECDRR+DER
      IF(DER.GE.20.0)DER=20.0
      IF(DER.LT.-35.0)DER=-35.0
```

```
* DECDRL
  433 DECDRL=((DECMD+DACMD)-DEL)*20.0
      IF(DECDRL.GE.20.0)DECDRL=20.0
      IF(DECDRL.LE.-20.0)DECDRL=-20.0
* DEL
  440 DEL=.04*DECDRL+DEL
      IF(DEL.GT.20.0)DEL=20.0
      IF(DEL.LT.-35.0)DEL=-35.0
*   DE AND DA
      DE=(DEL+DER)/2.0
      DA=(DEL-DER)/2.0
* DSBC
  450 DSBPC=SBHP-DSBC1
      IF(DSBPC.GT.6.1) DSBPC=6.1
      IF(DSBPC.LT.-10.86) DSBPC=-10.86
      DSBC1=DSBC1+DSBPC
      DSBC=.5*DSBC1
* DRCDRR
      DRCDRR=((DRCMD-DSBC)-DRRP)*10.0
      IF(DRCDRR.GE.10.0)DRCDRR=10.0
      IF(DRCDRR.LT.-10.0)DRCDRR=-10.0
* DRRP
  460 DRRP=.04*DRCDRR+DRRP
      IF(DRRP.GT.54.88) DRRP=54.88
      IF(DRRP.LT.-54.88) DRRP=-54.88
* DRCDRL
  463 DRCDRL=((DRCMD+DSBC)-DRLP)*10.0
      IF(DRCDRL.GE.10.0)DRCDRL=10.0
      IF(DRCDRL.LT.-10.0)DRCDRL=-10.0
* DRLP
  470 DRLP=.04*DRCDRL+DRLP
      IF(DRLP.GT.54.88) DRLP=54.88
      IF(DRLP.LT.-54.88) DRLP=-54.88
* DR
      DR=(DRRP+DRLP)*0.5
* DSB
      DSB=DRLP-DRRP
* DBFRC
      IF(DBFRC.GE.3.0)DBFRC=3.0
      IF(DBFRC.LE.-1.0)DBFRC=-1.0
* DBF
  490 DBF=DBF+0.04*DBFMAN
      IF(DBF.GE.22.5)DBF=22.5
      IF(DBF.LT.-11.7)DBF=-11.7
      RETURN
*
*   THE FOLLOWING SECTION INITIALIZES ALL INTEGRATOR OUTPUTS AND
*   INTERMEDIATE TRANSFER FUNCTION NODES TO ZERO.
*
 1000 DETR=ETRIM=ELFBK=0
      PROD25=DRTMSF=DRTRIM=0
      DATSUMI=0.0
*   THE TRANSFER FUNCTION NODES ARE INITIALIZED TO ZERO HERE.
*
      DETRN=0.0
      ETRIMN=ELFBKN=0
      PROD25N=DRTMSFN=DRTRIMN=0.0
      DATSUMN=0
      GO TO 3
```

```
      END
*************************************************************************
*                                                                       *
*                               F I L T 1                               *
*                                                                       *
*************************************************************************
      FUNCTION FILT1(XIN,GX1,GX2,GX3,XNODE)
      FILT1=XNODE+XIN*GX1
      XNODE=XIN*GX2-FILT1*GX3
      RETURN
      END
*************************************************************************
*                                                                       *
*                               F I L T 2                               *
*                                                                       *
*************************************************************************
      FUNCTION FILT2(XIN,GX1,GX2,GX3,GX4,GX5,XNODE1,XNODE2)
      FILT2=XIN*GX1+XNODE1
      XNODE1=XIN*GX2-FILT2*GX4+XNODE2
      XNODE2=XIN*GX3-FILT2*GX5
      RETURN
      END
*************************************************************************
*                                                                       *
*                             H Y S T E R                               *
*                                                                       *
*************************************************************************
      SUBROUTINE HYSTER (LIM1,LIM2,ARGT0,ARGT1,FUNCT)
      REAL LIM1,LIM2
*
*     THIS SUBROUTINE PERFORMS THE HYSTERESIS FUNCTION USING
*    PARAMETERS FROM THE CALLING PROGRAM.  THE ARGUMENTS ARE AS FOLLOWS:
*
*
*          LIM1 AND LIM2 ARE THE POSITIVE BREAK POINT VALUES.
*          (LIM1 LESS THAN LIM2). IT IS ASSUMED THAT THE FUNCTION
*          IS SYMETRICAL ABOUT THE ORIGIN.
*
*          ARGT0 AND ARGT1 ARE THE INPUT VALUES FOR THE TWO MOST
*          RECENT TIME FRAMES WITH ARGT1 THE MOST RECENT. THE CALLING
*          ROUTINE MUST CORRECTLY UPDATE THESE VALUES BEFORE CALLING
*          HYSTER.
*
*          FUNCT IS THE OUTPUT VALUE RETURNED TO THE CALLING PROGRAM.
*          FUNCT IS SET TO -1, 0, OR +1
*
*
        TEMP1  =  ABS(ARGT1)
      IF (TEMP1 - LIM1) 10,20,20
   10 FUNCT = 0.0
  900 RETURN
   20 IF ( TEMP1 - LIM2 ) 40,30,30
   30 IF ( ARGT1) 60,50,50
   50 FUNCT  =  1.0
      GO TO 900
   60 FUNCT  =  -1.0
      GO TO 900
   40 IF ( ARGT1) 80,70,70
*
```

```
*     COME HERE IF INPUT IS BETWEEN -LIM1 AND -LIM2.
*     CHECK PREVIOUS INPUT VALUE AND SET FUNCT ACCORDINGLY OR
*     LEAVE IT UNCHANGED.
*
   80 IF ( ARGIO + LIM1 ) 900,900,10
*
*     COME HERE IF INPUT IS BETWEEN LIM1 AND LIM2.
*     CHECK PREVIOUS INPUT VALUE
*
   70 IF ( ARGIO - LIM1 )   10,900,900
      END
*******************************************************************
*                                                                 *
*                      F U N C T I O N                            *
*                                                                 *
*******************************************************************
C 'FUNCTION' IS USED TO DETERMINE THE OUTPUT OF A FUNCTION SCHEDULE THAT
C     HAS MINIMUM,MAXIMUM, AND INTERMEDIATE LINEAR VALUES.  THE FUNCTION
C     SCHEDULE CAN HAVE A MAXIMUM OF TWO BREAKPOINTS.  TABLE(3) IS THE
C     SMALLEST ARGUMENT BREAKPOINT VALUE AND TABLE(4) IS THE LARGEST.
C     TABLE(1) IS THE OUTPUT VALUE FOR THE CORRESPONDING TABLE(3) (SMALL)
C     ARGUMENT AND TABLE(2) IS THE CORRESPONDING VALUE FOR THE TABLE(4)
C     (LARGE) ARGUMENT.
      FUNCTION FUNCTION(ARG,TABLE)
      DIMENSION TABLE(4)
      IF(ARG.GT.TABLE(3))GOTO3000
      FUNCTION=TABLE(1)
      GOTO 3002
 3000 IF(ARG.LT.TABLE(4))GOTO 3001
      FUNCTION=TABLE(2)
      GOTO 3002
 3001 FUNCTION=TABLE(1)+(ARG-TABLE(3))*((TABLE(2)-TABLE(1))/(TABLE(4)-
     *TABLE(3)))
 3002 RETURN
*******************************************************************
*******************************************************************
**                                                               **
**                                                               **
**                                                               **
*******************************************************************
*******************************************************************
      END
```